UNCLASSIFIED

AD NUMBER
ADB009952
LIMITATION CHANGES
TO: **Approved for public release; distribution is unlimited.**
FROM: **Distribution authorized to U.S. Gov't. agencies only; Test and Evaluation; OCT 1974. Other requests shall be referred to Army Ballistic Missile Defense Advanced Technology Center, Attn: ATC-2, PO Box 1500, Huntsville, AL 35807.**
AUTHORITY
bmdsc, d/a ltr, 13 dec 1976

THIS PAGE IS UNCLASSIFIED

SURC TR 75–249 (Volume 6)

ECM/ECCM Evaluation Program

Large Aperture Adaptive Array Study
Final Report

D.J. Chapman
S.G. Kamak

Prepared for
U.S. Army
Ballistic Missile Defense Advanced Technology Center
Huntsville, Alabama

March 1976

SURC
SYRACUSE UNIVERSITY RESEARCH CORPORATION

The findings of this report are not to be construed as an official Department of the Army position.

11

SURC-TR-75-249-VOL[redacted]6)
Mar[redacted]976

11

Copy 6 of 34

6

ECM/ECCM EVALUATION PROGRAM. Volume 6.

LARGE APERTURE ADAPTIVE ARRAY STUDY.

9 FINAL REPORT. 29 Jan 74-28 Feb 76,

10 Dean J. Chapman
Stephen G. Kamak

12 91p.

Prepared for

U.S. Army Ballistic Missile Defense Advanced Technology Center
Huntsville, Alabama

15

Contract	DAHC60-73-C-0044	
Effective Date of Contract	29 January 1973	
Contract Expiration Date	28 February 1976	
Amount of Contract	$1,474,488.00	
Principal Investigator	D. J. Chapman	(315) 477-8759
Project Engineer	D. P. Morezak	(315) 477-8495

Distribution limited to U.S. Government Agencies only; Test and Evaluation; 4 October 1974; other requests for this document must be referred to Ballistic Missile Defense Advanced Technology Center, ATTN: ATC-2, P.O. Box 1500, Huntsville, AL 35807.

339 750

FOREWORD

The effort described in this report was performed by Syracuse University Research Corporation (SURC) under Contract DAHC60-73-C-0044 sponsored by the U.S. Army Ballistic Missile Defense Advanced Technology Center. This final report covers the period from 29 January 1974 through 28 February 1976.

This report is identified by the contractor as SURC TR 75-249. Volume 6 is submitted as part of the final report, entitled *ECM/ECCM Evaluation Program*, which comprises the following eleven volumes:

Volume	Title
1	Summary
2	ECCM Effectiveness Evaluation
3	Distributed Array Radar Study
4	Row-Column Precision Adaptive Array Study
5	ECM Threat Assessment
6	Large Aperture Adaptive Array Study
7	Hybrid Loop Design Study
8	Dome Antenna Investigations
9	Precision Considerations in Large Adaptive Arrays
10	Precision Frequency Scanned Arrays for Terminal Defense Systems
11	Phased Array Radar Cost Optimization Study

ABSTRACT

This report is concerned with the problem of protecting a large phased array radar from the effects of electronic countermeasures (ECM). The context of the problem is the defense of strategically important targets from ballistic missile attack, the radar being one of the key sensors in the defense system.

A continuation of previously reported work, this study investigated the feasibility of using a partially adaptive array to emulate, as nearly as possible, the performance of the fully adaptive array. A partially adaptive array would be less expensive to fabricate and less complex and, therefore, would be capable of more rapid response.

This study was divided into three parts. First, since the partially adaptive array can be implemented in many different configurations, several of the more promising of these configurations were investigated. Second, the fundamental question of transient response was addressed. The findings of this portion of the study apply to any adaptive processor. A potential solution to the transient response problem was postulated and evaluated. Since this solution, which primarily addressed problems arising during the radar search function, had a potential effect on the performance of the radar during the track function, a third and much shorter study was launched into the question of tracking performance of the array.

The results of these studies have shown that there are several partially adaptive array configurations which show promise of providing the protection needed against sidelobe jamming. Further, response time of these arrays can be controlled to within acceptable limits. There does not, however, appear to be a simple solution to the requirement for precision in the fabrication of the partially adaptive array. Array errors would seem to pose a fundamental limitation to this approach.

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

LIST OF ILLUSTRATIONS

LIST OF ILLUSTRATIONS (Continued)

SECTION 1

INTRODUCTION

This report is concerned with the problem of protecting a large phased array radar from the effects of electronic countermeasures (ECM). The context of the problem is the defense of strategically important targets from ballistic missile attack, the radar being one of the key sensors in the defense system.

As time has passed, the techniques for negating the effects of jamming have evolved from relatively elementary sidelobe cancellation systems to the concept of a fully adaptive array, keeping astride the ever-increasing postulated threat. As the fully adaptive array concept began to emerge, it soon became obvious that making a large phased array fully adaptive would be expensive as well as potentially too slow to respond to rapidly changing input conditions.

This study, a continuation of previously reported work[1], investigated the feasibility of using a partially adaptive array to emulate, as nearly as possible, the performance of the fully adaptive array. A partially adaptive array would be less expensive to fabricate and less complex and, therefore, would be capable of more rapid response.

This study was divided into three parts. First, since the partially adaptive array can be implemented in many different configurations, several of the more promising of these configurations were investigated. Second, the fundamental question of transient response was addressed. The findings of this portion of the study apply to any adaptive processor. A potential solution to the transient response problem was postulated and evaluated. Since this solution, which primarily addressed problems arising during the radar search function, had a potential effect on the performance of the radar during the track function, a third and much shorter study was launched into the question of tracking performance of the array.

Section 2 of this report presents a summary of the results and conclusions which evolved from this study. Sections 3 and 4 present the partially adaptive array configuration studies and the performance limitations imposed by array errors. Sections 5 and 6 address the adaptive processor transient response question, while Section 7 discusses the impact of a particular processor configuration upon the tracking performance of the radar system. Appendices A through H present supportive information.

SECTION 2

RESULTS AND CONCLUSIONS

2.1 RESULTS

During this investigation, several partially adaptive configurations applicable to planar arrays were analyzed. Each was evaluated on its ability to counteract sidelobe jamming.

The first configuration checked was the so-called simple subarray. This consisted merely of groupings of contiguous elements summed to form the input to the adaptive processor. Whenever such subarrays are formed, nonuniform weighting at the subarray level produces echelon lobes at regular intervals in the sidelobe region. To prevent this from happening, it it necessary to ensure that the combined effects of the element pattern and the subarray pattern produce little or no gain in these directions. This usually means fixed, nonuniform tapering on each subarray.

Another technique particularly suited to series-fed arrays is the row (or column) subarray. Here, the row is preweighted to yield low sidelobes in one dimension. By adaptively weighting the row outputs, the sidelobes in the other dimension can be controlled. The disadvantage of this technique is that the taper loss and beam broadening caused by the open-loop weighting of the rows also penalizes the peacetime (non-ECM) performance.

A technique which alleviates this problem at the expense of additional beam forming hardware is the so-called row-column subarray configuration. Each element is divided into two paths. One path is used to form preweighted row subarrays with adaptively weighted outputs. The other path is used to form column subarrays which are weighted adaptively also. All row and column outputs are then summed to form the beam. The advantages of this technique over the previous one are that: (1) adaptive control is maintained in both dimensions (azimuth and elevation), and (2) less severe tapering on each row and column is needed, yielding better peacetime performance. The reason for the latter is that only -30 dB tapering is required for -60 dB sidelobes in the regions off the principal axes, since the resulting pattern is the product of the two weighting functions. For the row-only configuration, -60 dB weighting is required to obtain -60 dB sidelobes.

All configurations were simulated on a digital computer and were found to perform in a satisfactory fashion, provided that array errors were maintained at a negligible level. Once array error levels were allowed to increase to levels consistent with current state-of-the-art, performance was limited to sidelobe levels consistent with the theory associated with random array errors in the non-adaptive array. More precisely, the fully adaptive array, which is insensitive to this class of error, is also insensitive to the specific number of jammers in its sidelobes. The partially adaptive array with high error levels becomes sensitive to jammer number in a fashion not unlike the ordinary multiple sidelobe canceller.

The bounds on the transient response of an adaptive processor are imposed partly by the statistical properties of the input signals and partly by the adaptive algorithm employed and the way in which it is implemented. In Section 5, two major classes of adaptive algorithms are presented, and their relative merits are discussed as they relate to the transient response problem.

The theoretically optimum solution to the adaptive array problem produces a set of array weights which is a function only of the desired signal vector and the noise covariance matrix. The first algorithm discussed is called the direct matrix algorithm. The true covariance matrix, from which the ideal weights are computed, involves an infinite time average of the input process. The direct matrix algorithm uses a sample covariance matrix which is an average over a finite number of (independent) samples of the input process. It is shown that, for an N-channel system, only about 2N samples are needed to approach within 3 dB of the theoretical optimum performance.

Unfortunately, the computations required to implement this algorithm make it difficult to use in a high-speed real-time environment. A special-purpose vector arithmetic unit is needed.

The steepest descent (or control loop) algorithm, on the other hand, is easily implemented for high-speed applications in either digital or analog form. This algorithm, based upon a well-known iterative minimization technique, employs a closed feedback loop and is, therefore, self-correcting in terms of processing errors. The disadvantage of this algorithm is that the convergence time (the time it takes the processor to reach an acceptable solution) is difficult to predict a priori and is, in general, subject to a wider variation than the direct matrix algorithm.

A numerical example worked out in Section 5 shows that a 100-channel system, operating at a signal bandwidth of 10 MHz and using special-purpose vector multiplier hardware, takes about 6 ms to find a solution using the direct matrix algorithm. This solution yields an SNR which is within 3 dB of the theoretical optimum. The steepest descent processor would take between 1 and 100 μs to settle to approximately 20 dB of noise cancellation.[†]

There are two observations which can be made concerning this example. The time taken by the direct matrix algorithm is predominately due to computations required and not to accumulating sufficient samples of the input. At this bandwidth, the 100-channel processor would require only about 20 μs to accumulate sufficient samples. The steepest descent processor, on the other hand, performs the simple computations it needs during each intersample period. Its convergence time is strictly limited by the number of independent samples required and the rate at which those samples are available. Second, the comparison between the two algorithms is not exactly equivalent since the method of analysis and the criterion used in each case is different. More theoretical analysis (Appendix H) is needed before the two algorithms can be compared on a more equivalent basis.

[†] Timing in this example is based upon a single-pole filter. Higher order filters will shorten this response by up to 4:1.

Nevertheless, it is clear that, for large-dimension, high-speed, real-time applications, the steepest descent (or loop-type) processor is capable of reaching a solution starting with zero initial conditions faster than the direct matrix algorithm.

In Section 6, attention is given to the specific problem of the transient response of the large aperture partially adaptive array. For the ballistic missile defense radar, a serious threat is posed by the precursor jammer. A typical threat consists of several hundred parachuteborne jammers slowly descending on the area defended by the radar.

Ordinarily, the adaptive array would be expected to settle down quickly each time the beam is switched to a new position. A scheme was evaluated wherein weights for each beam position were stored in a memory and recalled as needed. They were updated each time the beam was switched to that position. The critical question concerning the feasibility of this approach is the relative stationarity of the jammer noise field compared with the processor update interval (that is, how often each beam position is visited). The results of a simulation detailed in Section 6 show that, for typical parachute descent rates and 1 to 3 s frame times, the system performs nicely. Using the relatively long-term stationarity of the jammer field allows one to use either algorithm.

The technique outlined in the previous paragraph results in the search volume being divided into discrete beam positions. This is acceptable for the search function. However, normal tracking procedure, which results in the track beam always being centered on the object being tracked, would require many more overlapping beam positions than are stored. The effects upon tracking performance of making use of the available discrete beam positions and the use of "off-null" tracking were investigated. An existing tracking simulation program was used to evaluate these effects as a function of the target signal-to-noise ratio (SNR). Very little effect was noticed at low and high SNRs. Noticeable degradation did occur for moderate signal strength, however. It was concluded that this degradation could be avoided by suitably altering the tracking algorithm, particularly in the track initiation phase.

2.2 CONCLUSIONS

In terms of protecting the sidelobes of an antenna from directional interference, there are three catagories one might use to catalog adaptive systems, the distinction being made according to complexity and performance. First, there is the sidelobe canceller. This could be thought of as a system which controls a very small percentage of the total degrees of freedom needed to completely specify the far-field pattern of the array. This system is designed to handle a few discrete sources of interference. The performance is a function of the ratio of independent sources to adaptive degrees of freedom, as well as instantaneous bandwidth and other effects.

At the other end of the spectrum is the fully adaptive aperture in which the number of adaptive degrees of freedom employed is sufficient to completely specify the far-field pattern. The performance is essentially independent of the number of discrete sources since the entire sidelobe structure can be adjusted more or less at will.

Finally, there is the partially adaptive array which may have a leverage of from 10:1 to 50:1 in the ratio of total degrees of freedom to adaptive degrees of freedom. This is a transitional case and the performance is transitional as well. As the results have indicated, certain configurations are capable of performance approaching that of the fully adaptive array provided that error levels are sufficiently low. For those cases in which errors are high, the partially adaptive array is limited by those errors for large numbers of jammers and behaves like a sidelobe canceller for small numbers of sources. In other words, unless the ratio of jammers to adaptive degrees of freedom is low, the partially adaptive system with high errors will not provide good cancellation.

With the proliferated sidelobe jamming threat, a point is reached when a significant number of beam positions contain one or more jammers posing a mainlobe jammer threat. Both the partially and fully adaptive array will provide some measure of mainlobe protection, depending upon the types of constraints employed. However, once the spatial density of jammers reaches a certain point, there is no alternative but to increase the size of the aperture. The adaptive array will not increase resolution above that which a given size aperture is capable.

The results of these studies have shown, therefore, that there are several partially adaptive array configurations which show promise of providing the protection needed against sidelobe jamming. Further, response time of these arrays can be controlled to within acceptable limits. There does not, however, appear to be a simple solution to the requirement for precision in the fabrication of the partially adaptive array. Array errors would seem to pose a fundamental limitation to this approach.

SECTION 3

ARRAY CONFIGURATION STUDIES

In considering the various candidate subarray configurations, the first that comes to mind is the grouping of physically contiguous elements to form what shall be termed "simple subarrays." The usual technique for analyzing this configuration is to factor the radiation pattern function into an array factor and a subarray factor. Using this concept, it can be seen that the adaptive processor is controlling the array factor and the "elements" of this array are, in fact, the subarrays. The immediate problem with this configuration is that the phase centers of the subarrays are usually separated by several wavelengths, with the result that the array factor repeats several times in visible space. The echelon lobes thus generated, while modified by the subarray pattern to a certain degree, cannot be altered further by the adaptive processor because the folding-over process has caused the echelon lobes to become indistinguishable from the main beam as far as the array factor is concerned. A scheme was investigated(2) for providing additional open loop suppression of echelon lobes by placing multiple nulls of the subarray pattern in the directions of the echelon lobes. However, in order to obtain enough control of the subarray pattern to accomplish this, overlapping subarrays were used. This approach is feasible subject to the error limitations discussed in Section 4. Unfortunately, the beamforming network for this approach is complicated and somewhat inefficient.

One of the more attractive configurations studied is the "beam-space" subarray. The term subarray tends to be misleading in this case because the full aperture is used for each subarray. It could also be called a multibeam processor. In its simplest form, the beam-space configuration consists of a linear array of n elements (where n is preferably an integer power of 2) connected to a Butler(3) matrix beamformer. This beamformer normally provides n beams simultaneously at the output side, but this configuration uses only the beam in the desired scan direction and several others immediately around it. Figure 4-1(d) shows an example of the performance of a 32-element, 8-channel beam space array.

One of the most straightforward applications of the beam-space concept to planar arrays is in the space-fed array. In this type of array, a cluster of feed horns in the focal plane of the array lens provides a cluster of beams in two dimensions similar to the previous example for the linear array. Each horn would be connected to one channel of the processor. Various other schemes have been proposed in the past for forming multiple simultaneous beams, and any of these could also be used in this way. Results of previous simulations(1) show the beam-space configuration to have better transient response behavior than other configurations studied, when used in conjunction with a loop-type processor. The beam-space adaptive array is, of course, subject to the same limitations due to array errors as the other subarray techniques.

The existence of planar array designs employing slotted waveguide or other series-feed configurations involving rows of elements suggested the adaptive combining of row (or column) subarrays. It was realized at the outset that this would yield adaptivity in only one principal plane of the far-field pattern, however. Control along the other principal plane must be accomplished in an open-loop fashion by putting the proper fixed amplitude taper on the elements in each row. Unfortunately, this usually requires rather heavy fixed tapering to anticipate worst-case jamming conditions. This affects the efficiency of the array under peacetime conditions. Nevertheless, Figure 3-1(a) depicts the quiescent pattern of a 100-element array arranged in a square planar array 10 elements on a side. Each 10-element row is weighted for 40 dB Dolph-Tchebychev sidelobes. The response in the other principal plane corresponds to a uniformly weighted steering vector. Figure 3-1(b) shows the response to a concentrically placed ring of interference sources adjusted to cut through the principal planes midway through the sidelobe region. Note that the response is only in the one principal plane, as expected.

The need for heavy open-loop weighting and lack of control in one principal plane in the row-only configuration led to the next configuration, termed the row-column precision adaptive array (RCPAA).(4) Figure 3-2 illustrates how this system is implemented. Each element signal is split into two paths. One path goes to a row summer as in the row-only array described previously. The

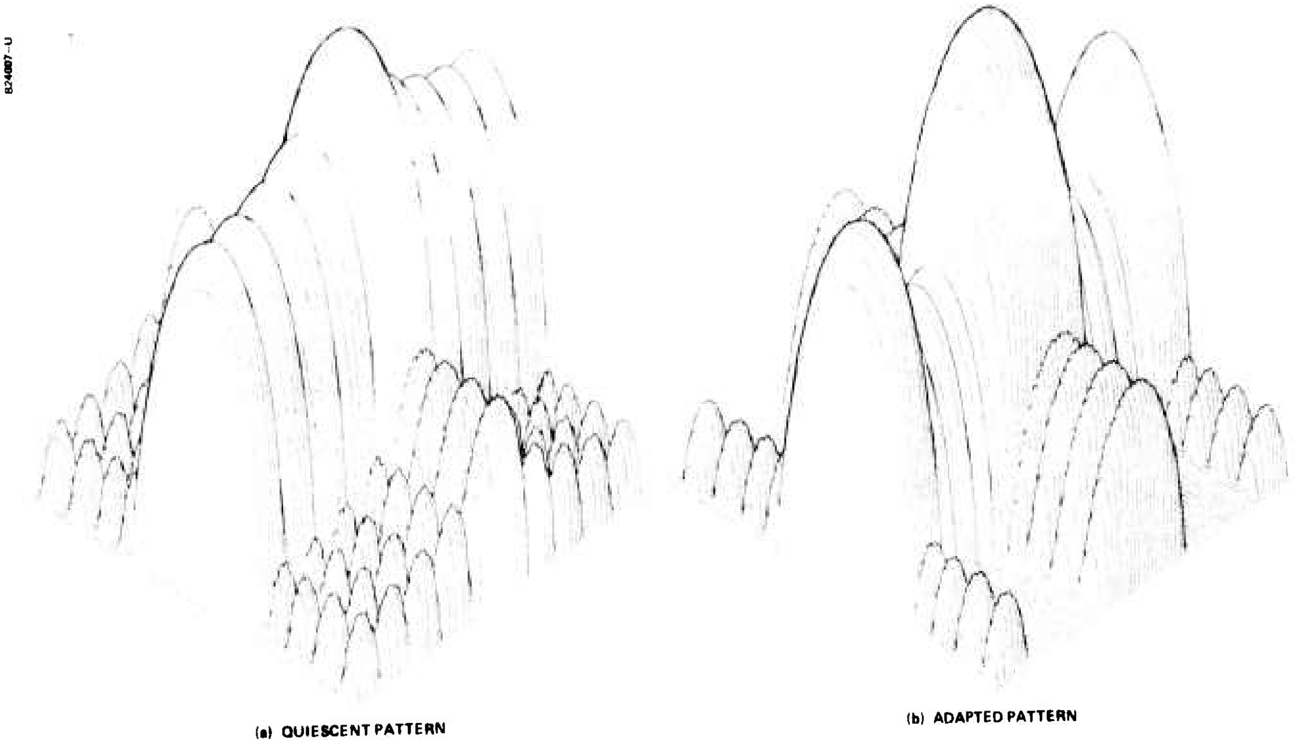


Figure 3-1. Row Subarrays

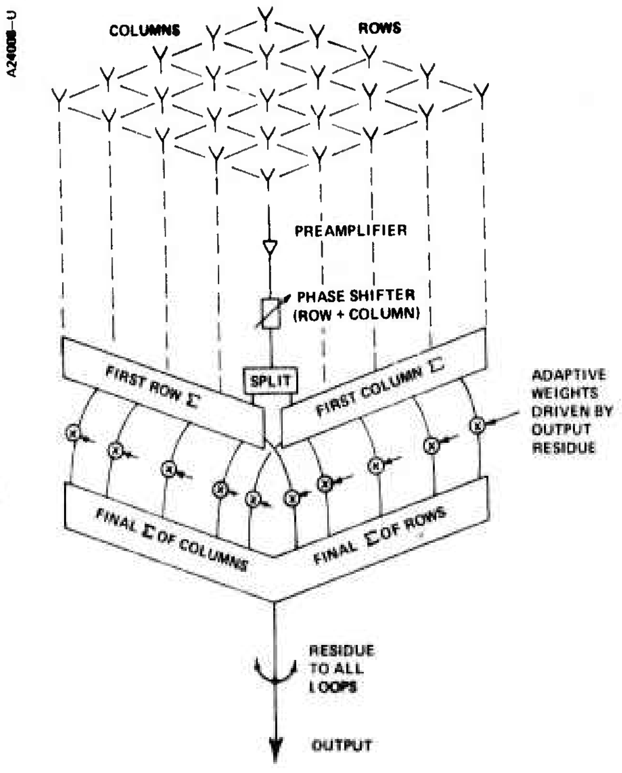
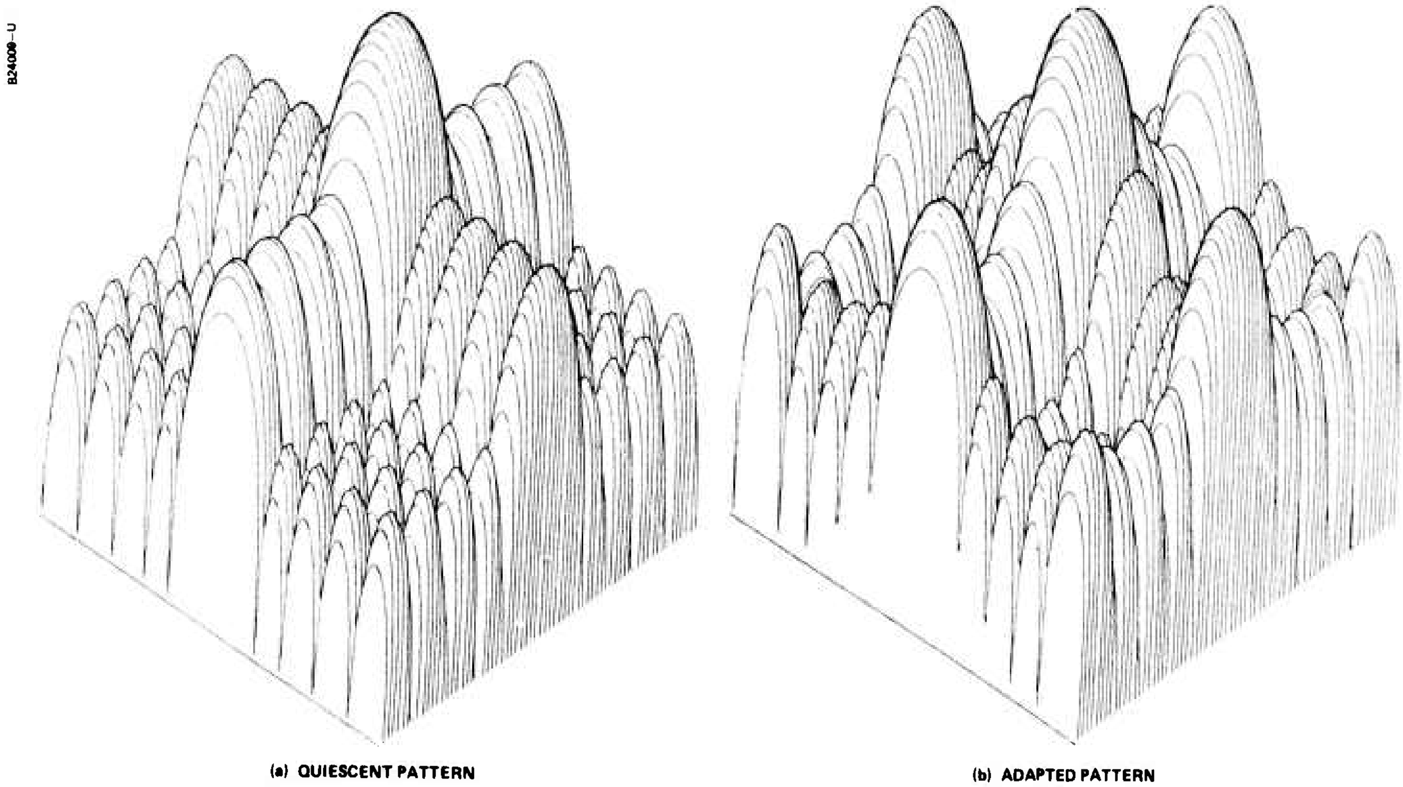


Figure 3-2. Precision Adaptive Array

rows are combined adaptively. The other half of the element power is fed to a column summer. The column outputs are then combined adaptively at the same point as the row outputs. The number of adaptive degrees of freedom, is, therefore, equal to the number of rows plus the number of columns in the array.

Figure 3-3(a) illustrates the quiescent pattern for this configuration using the same 100-element geometry as the previous example. Each row or column is weighted with 23 dB Dolph-Tchebychev coefficients. Note the product effect on the off-axis sidelobes. The use of less severe weighting means that the peace-time efficiency of the array is relatively high. In the presence of jamming, the response along a principal plane will be the sum of a fixed pattern due to the fixed weighting in one direction and an adaptive pattern due to the adaptive weighting in the orthogonal direction. This configuration can be thought of as a form of beam-space processor where each beam is formed by a row or column. Primary control is exercised in the principal plane sidelote areas.



(a) QUIESCENT PATTERN

(b) ADAPTED PATTERN

Figure 3-3. Row-Column Array

Because of the product effect, off-axis sidelobes are much lower initially and do not need the same degree of protection. Figure 3-3(b) illustrates the response of this array to the same concentric ring of interference sources as employed in the previous example. Note the response along each principal plane. Incidentially, isotropic element patterns were employed throughout, which accounts for the large sidelobes near endfire.

The advantages of this configuration have been outlined. The primary disadvantage of this configuration is a practical one of implementation. A straightforward corporate feed structure which not only provides the redundant row and column sums needed for the sum pattern but also provides two-axis monopulse difference beams as well could be cumbersome. Again, space feed techniques could prove to be more practical from a hardware standpoint.

SECTION 4

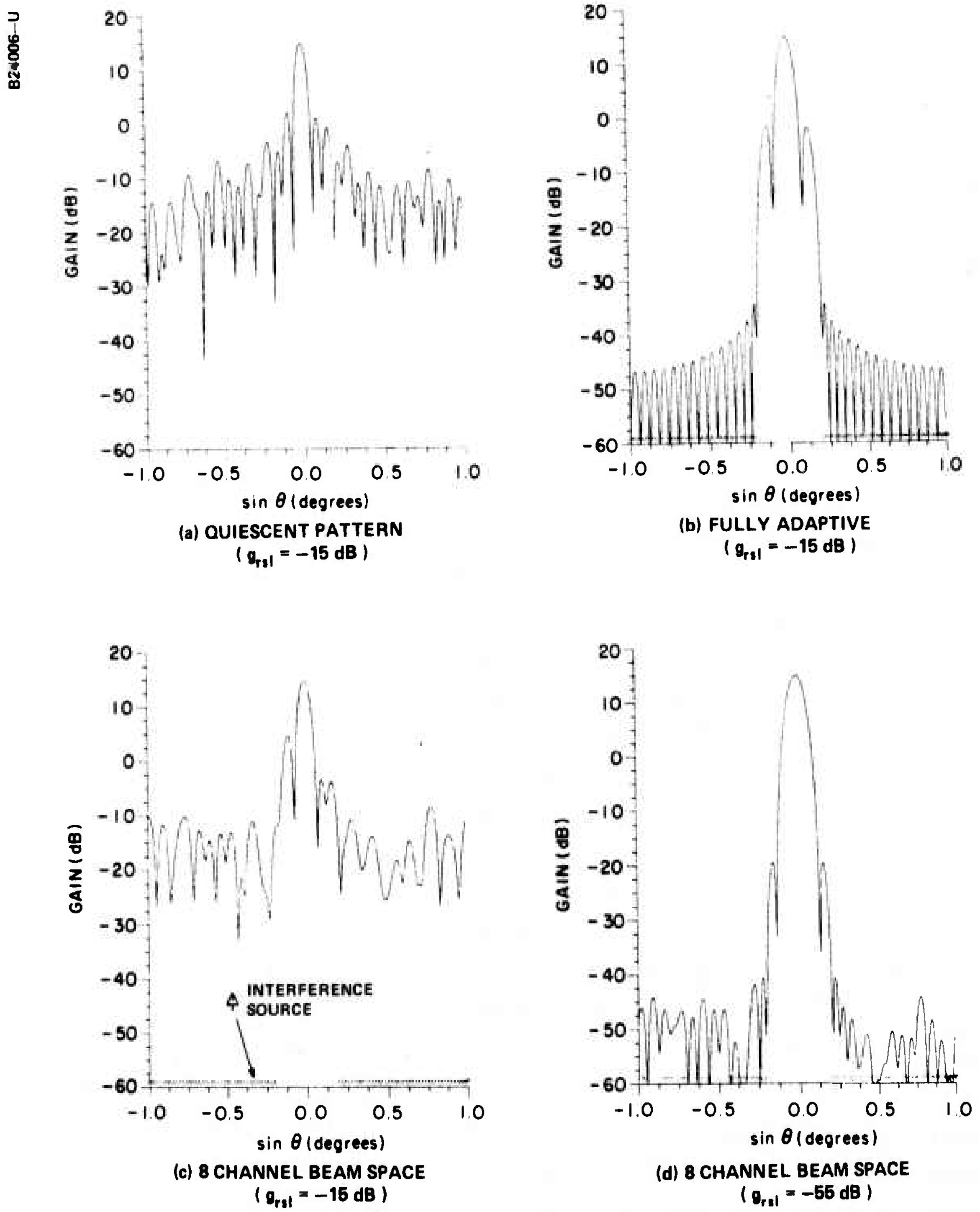
ARRAY ERROR LIMITATION STUDIES

Under ideal conditions, most subarray configurations studied were capable of yielding performance nearly equal to that of fully adaptive arrays, but with a fraction of the adaptive degrees of freedom of the fully adaptive array. This is made possible by the fact that the subarray transformation "folds" real element n-space into a subarray m-space by taking advantage of symmetries in the configuration. Thus, m degrees of freedom at the subarray level appear to have the same effect as n degrees of freedom at the element level through the leverage of the subarray beamformer.

However, the introduction of random independent errors at the element level destroys the needed symmetry. The adapted radiation pattern of an array with such errors can be thought of as consisting of a deterministic component controlled by the adaptive processor and a random component determined by the level of errors in the array. Under severe interference conditions, the deterministic component of the sidelobe pattern is lowered to the degree necessary to otherwise cancel the effects of the interferers. However, since the total response of the array is the sum of the random and deterministic components, the random sidelobe structure ultimately dominates.

Figure 4-1 illustrates this effect for a 32-element linear array. Figure 4-1(a) depicts the quiescent (no jamming) pattern for the array assuming element-level errors equivalent to -15 dBi (15 dB below isotropic). In Figure 4-1(b), the response to a severe jamming attack with sources spread throughout most of the sidelobe region is shown for the fully adaptive case with the same level of errors as in Figure 4-1(a). In Figure 4-1(c), the 32 elements are combined into an 8-channel beam-space† processor with the same -15 dBi error level. Note that while the fully adaptive array is perfectly capable of coping with this class of error, the partially adaptive array lacks the necessary degrees of

†See previous description of beam-space configuration.



(a) QUIESCENT PATTERN (g_{rsl} = −15 dB)

(b) FULLY ADAPTIVE (g_{rsl} = −15 dB)

(c) 8 CHANNEL BEAM SPACE (g_{rsl} = −15 dB)

(d) 8 CHANNEL BEAM SPACE (g_{rsl} = −55 dB)

Figure 4-1. 32-Element Array

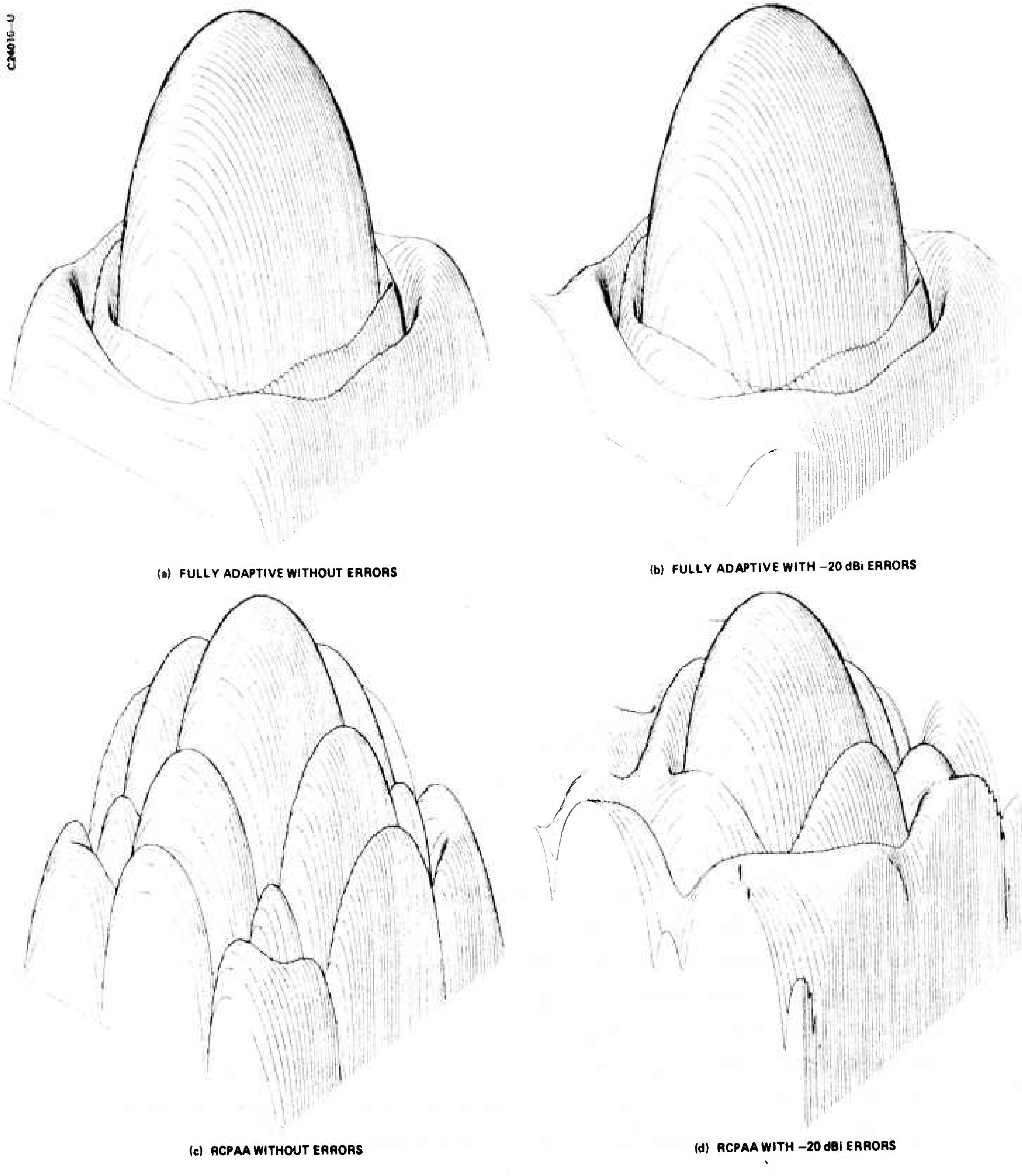
freedom to do so. If the errors are low enough, the beam-space processor works well, as in Figure 4-1(d) with -55 dBi errors.

To illustrate the effect of array errors on planar arrays, a 36-element array was configured in a square 6 elements on a side. The plot of Figure 4-2(a) shows the response of the fully adaptive processor to a set of concentric rings of interference with no array errors. For perspective, the floor of the plot is -60 dBi and the peak of the beam is +15 dBi. Figure 4-2(b) is the same fully adaptive case but with element errors equivalent to -20 dBi sidelobes. As expected, the fully adaptive array compensates for these errors.

In Figure 4-2(c), the 36-element array is configured as a row-column processor with 12 degrees of freedom. The same interference field is employed and errors are assumed negligible. Figure 4-2(d) is the same case but with -20 dBi errors. Because the interference field cuts the principal plane in only one region, the processor is able to cope with it. Note, however, that the off-axis sidelobe level is essentially that of the random component of the pattern. Therefore, this configuration needs low element-level errors to work well in a heavy jamming environment.

Other subarraying configurations might be considered, particularly if special circumstances exist, such as special hardware configurations or limited threat corridors. There are two characteristics which seem to be common to all of these approaches. One, of course, is the need for precision in the subarray beamforming in order to preserve as much of the symmetry as possible in the transformation. The other is the tendency of the subarraying process to create zones in space which can be adaptively controlled and those which have to be controlled by open loop means. In some configurations, such as row subarrays, this effect is very obvious. In others, such as the linear beam-space array, it is not as obvious. One of the most important considerations in choosing a configuration, therefore, is the need to minimize the effects of array tolerances on the region of the response which must be controlled in an open loop manner.

The effects of element-level errors on partially and fully adaptive arrays were summed up in a series of simulations summarized in Figure 4-3. A



(a) FULLY ADAPTIVE WITHOUT ERRORS

(b) FULLY ADAPTIVE WITH −20 dBi ERRORS

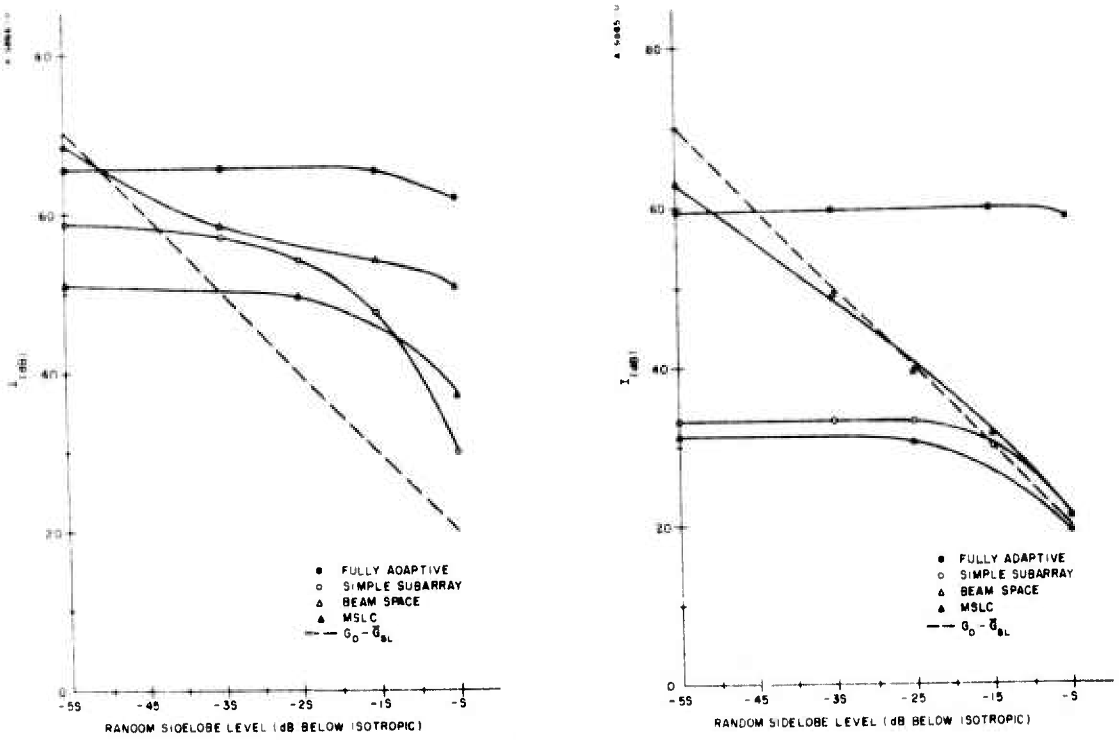
(c) RCPAA WITHOUT ERRORS

(d) RCPAA WITH −20 dBi ERRORS

Figure 4-2. Effects of Errors on Planar Adaptive Arrays

32-element linear array of half-wave-spaced isotropic elements was assumed. The frequency was 1300 MHz and the signal bandwidth was assumed to be 5 MHz. The jammer-to-noise ratio (JNR) in all cases was 60 dB in an element. The abscissa of the plots is the average sidelobe level due to assumed element-level errors. The ordinate is an improvement factor defined as the ratio of the signal-to-jammer-plus-noise ratio at the output of the array referenced to that same ratio in an element. Thus, this measure takes into account any loss in array gain due to errors as well as degradations in jammer cancellation. Figure 4-3(a) is a plot for seven discrete sources of jamming, and Figure 4-3(b) shows the results for 80 jammers covering most of the sidelobe region. Four adaptive configurations are compared. The reference case is a fully adaptive (32 adaptive degrees of freedom) configuration. The second case is the so-called simple subarray using eight degrees of freedom corresponding to eight 4-element subarrays. The third case is an 8-channel, beam-space configuration equivalent to the case described previously. Finally, the fourth case is a multiple sidelobe cancellation (MSLC) configuration where eight randomly chosen elements are designated an auxiliary elements. The dotted line shows the non-adaptive performance one would realize if the composite sidelobe level were that due to random errors only.

Note that, in Figure 4-3(a), the number of jamming sources is one less than the number of adaptive degrees of freedom. All configurations perform reasonably well. In Figure 4-3(b), complete control of the far-field pattern is needed to achieve usable performance. The fully adaptive configuration has the capability to do this and is relatively unaffected. The simple subarray is limited by the presence of echelon lobes which cannot be controlled by the adaptive array factor. The MSLC simply does not have the control needed. The beam-space configuration, through the folding-over process, is able to emulate fully adaptive performance only to the extent that the precision of the array allows. Note how closely it tracks the dotted line. It was noted in other simulations not shown here that the simple subarray configuration would exhibit this same behavior if one avoided placing jammers in the echelon lobe regions. This series of simulations illustrates the fact that partially adaptive arrays cannot arbitrarily control the random error component of the sidelobe structure, an impor-important limitation.



(a) 7 SOURCES

(b) 80 SOURCES

Figure 4-3. Performance Limitation due to Element-Level Errors on Partially and Fully Adaptive Arrays

SECTION 5

REVIEW OF MAJOR ADAPTIVE ALGORITHMS

The purpose of this section is to review two major classes of adaptive array algorithms with the purpose of quantifying the response time of each. Accordingly, this section begins with a short review of the theoretically optimum solution, introducing the notation to be used. There then follows a discussion of the direct matrix solution algorithm and, finally, a discussion of the steepest descent algorithm.

5.1 OPTIMUM SOLUTION

Figure 5-1 illustrates a canonical form of the adaptive array. The voltage, v_i, in the i'th element is weighted or multiplied by w_i and summed with the other weighted voltages to form the output. If V is the column vector containing all the v_i's, then the true covariance matrix is given by

$$M = E(V^*V^t) \qquad (5\text{-}1)$$

where the expectation is taken over (infinite) time. The asterisk denotes conjugation, and the t denotes matrix transpose. From considerations of maximum SNR(5), the optimum set of weights is given by

$$W_{opt} = aM^{-1}S^* \qquad (5\text{-}2)$$

where S is a vector of the voltages induced in each element by the desired signal and a convenient value for the coefficient, a, is

$$a = (S^tM^{-1}S^*)^{-1} \qquad (5\text{-}3)$$

5.2 DIRECT MATRIX SOLUTION AND IMPLEMENTATION

The closed form of the solution (Equation 5-2) for the optimum weights suggests the direct substitution of a best estimate of the matrix M; that is,

$$\hat{W} = a\hat{M}^{-1}S^* \qquad (5\text{-}4)$$

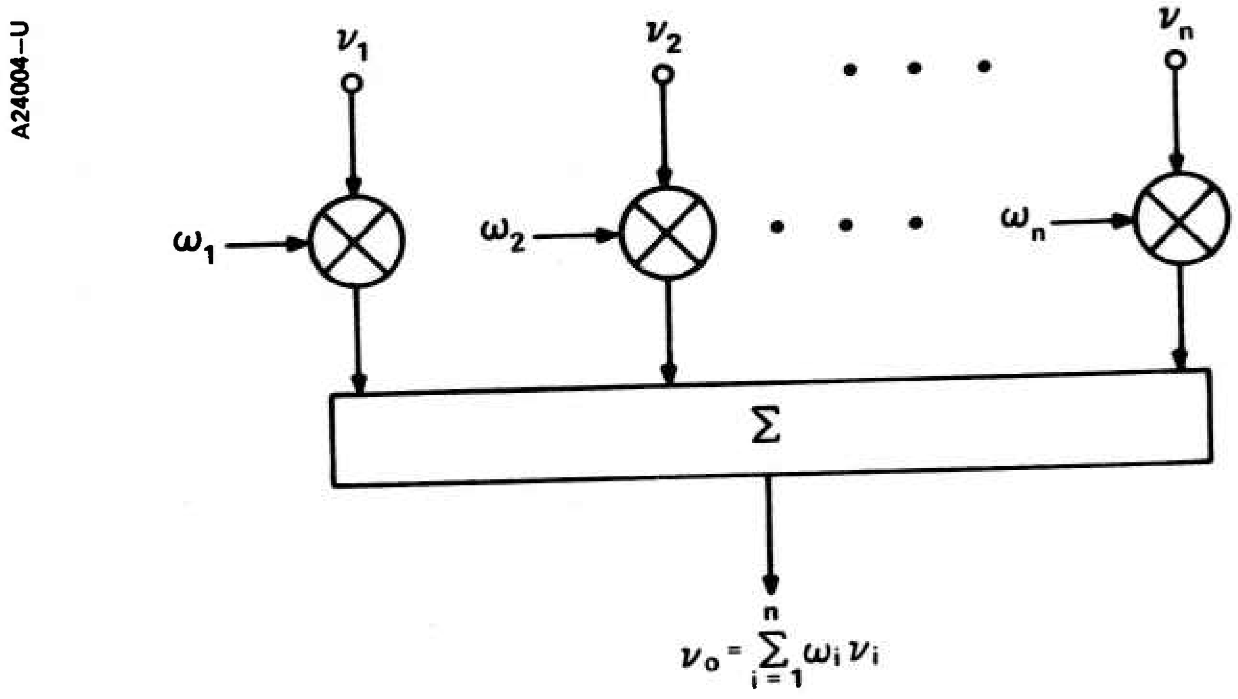


Figure 5-1. Canonical Adaptive Processor

where

$$\hat{M} = \frac{1}{K} \sum_{k=1}^{K} V_k{}^* V_k{}^t \qquad (5\text{-}5)$$

is an average over K samples of the input vector, V_k. For an N-element array, Reed[(6)] shows that the expected degradation from optimum SNR for finite K is given by

$$\text{loss} = -10 \log_{10} \frac{K + 2 - N}{K + 1} \text{ (dB)}, \ K > N \qquad (5\text{-}6)$$

or that approximately 2N samples will yield within 3 dB of the optimum SNR[†].

The direct matrix solution is, of course, an arithmetic solution and, as such, is implemented in digital hardware. The input voltages must be digitally encoded. The accuracy to which the encoding is carried depends upon the expected dynamic range of the signals. The encoding introduces quantization noise which must be treated just like any other source of noise in computing expected SNRs. The dynamic range of an encoded signal is given by the ratio of the power of a signal just short of saturation to quantization noise as

$$(S/N)_q = 6n + 5 \quad \text{(dB)} \qquad (5\text{-}7)$$

where n is the number of bits in the encoded word.

Because the adaptive process involves the subtraction of large numbers to form small but significant results (i.e., large jammer amplitude, small desired signal amplitude), it is very important to maintain the arithmetic accuracy of the calculations. The disadvantage of the direct solution is that it does not involve iteration or feedback and cannot, therefore, sense and correct computational errors. As a consequence, the arithmetic hardware employed in a real-time processor using the direct matrix algorithm must carry a large number of significant bits, a factor affecting both the cost and the speed of operation.

From Equation 5-5, the obvious procedure to follow would be to first accumulate the summation $V_k^* V_k^t$ to form the sample covariance matrix. Next, the

[†] Appendix F briefly summarizes the derivation of Equation 5-6.

matrix would be inverted using one of a number of standard matrix inversion techniques. In fact, this two-step process is not necessary in this case because of the special form of the matrix. It is possible to compute the inverse, $\hat{M}^{-1}$, directly from the data. Adding another sample of data to the sample covariance matrix according to the formula in Equation 5-5 is equivalent to

$$\hat{M}_{K+1} = \frac{K}{K+1} \hat{M}_K + \frac{1}{K+1} V_K^* V_K^t \tag{5-8}$$

Through the use of a matrix relationship (the derivation of which appears in Appendix E) one can alternately write

$$\hat{M}_{K+1}^{-1} = a\hat{M}_K^{-1} - (1 + bV_K^t \hat{M}_K^{-1} V_K^*)^{-1} \cdot c\hat{M}_K^{-1} V_K^* V_K^t \hat{M}_K^{-1} \tag{5-9}$$

where: $a = \frac{K+1}{K}$, $b = 1/K$, $c = \frac{K+1}{K^2}$.

To implement this form of the algorithm, one begins with a first estimate of $\hat{M}^{-1}$, which is simply

$$\hat{M}_1^{-1} = I \tag{5-10}$$

where I is the identity matrix of order N.

The form of averaging employed in Equations 5-5 and 5-8 is equivalent to the "integrate and dump" technique frequently used in digital processing. All samples in the data window of K samples receive equal weight in the average. It is also possible to implement the "sliding window" type of averaging process wherein the most recent sample receives the heaviest weighting and the previous samples are exponentially weighted in the fashion of a recirculating delay line with a gain slightly less than unity. The analog equivalent would be an RC low-pass filter. This form is particularly suited to quasistationary noise since it will adapt to a slowly changing environment automatically, gradually "forgetting" old data.

For the "sliding window" form of the process, the recursion takes the form of

$$\hat{M}_{K+1} = \rho M_K + (1 - \rho)V_K^* V_K^t \tag{5-11}$$

It can easily be seen then that the (K - J)th sample will have been weighted by ρ^J where ρ is a number slightly less than unity. To apply the matrix relationship of Equation 5-9, one uses the alternate definitions of the scaler constants

$$a = \frac{1}{\rho} , \ b = \frac{1 - \rho}{\rho} , \ c = \frac{1 - \rho}{\rho^2} \tag{5-12}$$

5.3 TIME ESTIMATE

In a nonreal-time application, a block of data would be processed by first forming the sample covariance matrix inverse, calculating the weights, and then using those weights to process the block of data. In a real-time system, one might use the sliding window equations and, for each new sample, apply Equation 5-9 (with the constants from Equation 5-12) once, then compute the new weights using Equation 5-4, and finally use the new weights on that sample, finishing in time to repeat the whole process with the next sample. Unfortunately, this schedule is very demanding of the arithmetic hardware, and is somewhat contradictory to the assumption which allows the formation of the sample covariance matrix in the first place; namely, that the noise field is at least quasistationary. In other words, the weights computed from one group of samples should hold for the next few samples with little or no degradation if parameters are changing very slowly. One could then take several sample intervals to perform the computations needed for each recursion.

It can be seen that the time required to obtain an estimate of the covariance matrix must be short with respect to the rate of change of the noise statistics, a fundamental limitation. If (see Equation 5-6) it takes about 2N samples to form a good estimate, and, from the constraint on computation time, it takes L sample periods to execute one recursion of Equation 5-9, then the minimum total computation time would be

$$T = 2NLt_s \tag{5-13}$$

where t_s is one sample interval. In practice, then, the noise field must change slowly with respect to T rather than to $2Nt_s$ in order for the real-time system to work properly and to keep up with the data input rate.

In order to apply the foregoing analysis to a specific case, it is necessary to investigate the bounds on the factor Lt_s. This, in turn, entails an estimate of the number and types of arithmetic operations required per recursion.

In a conventional general-purpose computer, arithmetic operations are done serially. Thus, when two complex numbers are multiplied and the result added to an accumulator, there are actually four scalar multiplication and four scalar addition operations performed. Using those ground rules, the number of multiplication operations involved in one recursion cycle of Equation 5-9 is $10N^2 + 6N$ and the number of addition operations is $8N^2 + 4N$. A typical third-generation computer (Xerox Sigma V) will execute a floating point multiplication in 10 μs and an addition in 5 μs. At this rate, it would take 0.5 s per iteration for a 100-element system or a T of about 5 min.

Obviously, a special-purpose machine is required. The predominant arithmetic operations involve N-element vectors with complex numbers for elements. Three fundamental operations are involved in the recursive cycle, vector addition, a vector times a scalar, and a vector inner product. There are N vector additions, N + 1 inner product operations and 2N + 1 vector/scalar multiplications. A vector arithmetic unit could be constructed which contains 4N scalar multipliers configured to carry out any of the above vector operations in parallel. It is within the state-of-the-art in present-day digital integrated circuits to carry out such an operation in 100 ns. For the 100-element processor used in the previous example, one cycle would take about 30 μs and would result in a T of about 6 ms (i.e., 2N recursion cycles).

It is significant to note for future reference that computation time, T, using a general purpose arithmetic unit is proportional to N^3, while it is proportional to N^2 for the vector arithmetic unit.

5.4 STEEPEST DESCENT ALGORITHM AND IMPLEMENTATIONS

The steepest descent or "control loop" algorithm is based upon the following development. For input voltage vector, V(t), and weight vector, W, the output residue voltage is

$$V_r(t) = W^t V(t) \tag{5-14}$$

The expected value of the residue power can be shown to be

$$P_r = W^H M\, W \tag{5-15}$$

where the $(\ )^H$ denotes conjugate transpose. This is a quadratic form and the N-dimensional surface described by this equation can be shown to be a concave hyperparaboloid. There is, therefore, only one minimum and no local minima. The steepest descent technique starts with an initial guess for W, computes the gradient at that point, and takes a small step in W along the direction of the negative gradient. The process is then iterated until there is no more improvement. Since the function is well-behaved over all W, a solution is always guaranteed.

That this method converges to the optimum can be easily shown by considering a simplified version of the development which appears in Appendix D. Suppose we wish to minimize the residue power (Equation 5-15) subject to the constraint

$$W^t S = 1 \tag{5-16}$$

That is, the response to the desired signal is always unity. This is merely a scale factor constraint. Let us form the cost function to be minimized as

$$F(W) = \frac{1}{2} W^H M\, W + \lambda\ (W^H S^* - 1) \tag{5-17}$$

where the factor $\frac{1}{2}$ and the conjugate version of Equation 5-16 have been substituted for convenience.

Taking the gradient of F(W) with respect to W, setting it to zero, and solving for W_{opt} yields

$$W_{opt} = \lambda\, M^{-1}S^{*} \tag{5-18}$$

Substituting the conjugate of Equation 5-18 into Equation 5-16 to solve for λ yields, finally,

$$W_{opt} = (S^{t}M^{-1}S^{*})^{-1}M^{-1}S^{*} \tag{5-19}$$

This result is essentially identical to the result obtained by Applebaum(5) from considerations of maximum SNR.

The steepest descent algorithm can be implemented with equal facility in either analog or digital form. The Howells(7) control loop for the sidelobe canceller and the Widrow(8) least mean square (LMS) algorithm are two early examples of, respectively, the analog and the digital realizations.

Considering first the digital form, Figure 5-2 shows a simple form of the control loop. Writing the equations for the loop, we have

$$W_i = S^{*} - X_i \tag{5-20}$$

and

$$X_{i+1} = X_i + k\, V_i^{*}V_i^{t}W_i \tag{5-21}$$

Combining these equations, one obtains

$$W_{i+1} = W_i - k\, V_i^{*}V_i^{t}W_i \tag{5-22}$$

The relevancy of this equation can be shown by taking the expected value of $W_{i+1} - W_i$ over time, yielding

$$E(W_{i+1} - W_i) = -k\, MW_i \tag{5-23}$$

That is, the expected value of the update to the vector W is proportional to the negative of the gradient of the residue function given by Equation 5-15. Thus, this system uses the instantaneous value of the gradient based upon the present data sample as an estimate of the true gradient. Since this is a closed feedback loop system, any errors made in one iteration (i.e., a step in a slightly less

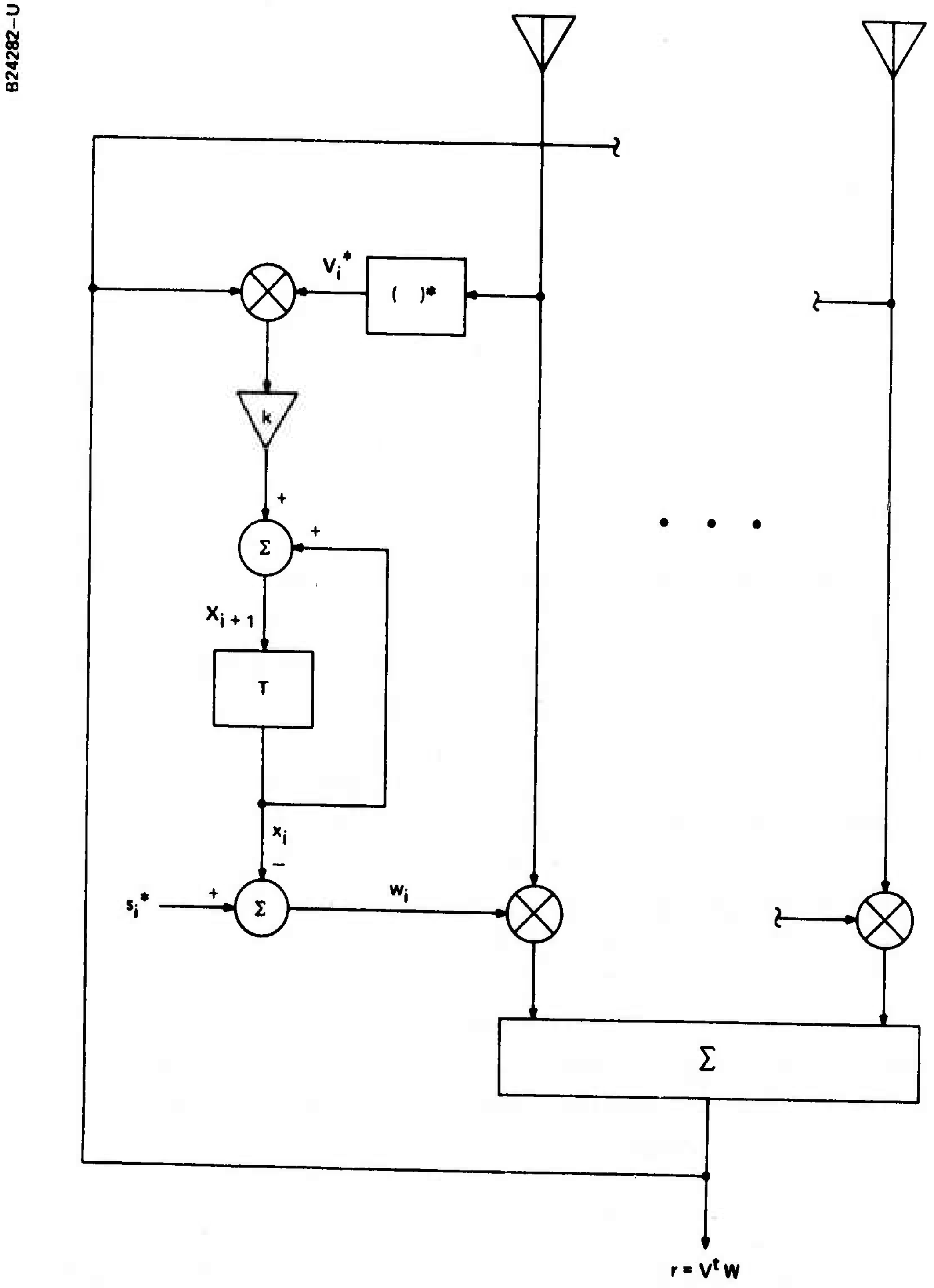


Figure 5-2. Simple Form of Digital Control Loop

than optimum downhill direction) will be corrected in the next iteration, a fact which eases the arithmetic precision requirements considerably.

With reference to Figure 5-3, the analog version of the control loop is only slightly different. The control equation for the weights is

$$W(t) = S^* - h(t)\ (*)\ k\ V^*V^tW(t) \tag{5-24}$$

where h(t) is the impulse response of a filter (usually a single pole low pass filter) and (*) denotes convolution. The filter, in essence, smooths the instantaneous estimate of the gradient similar to the action of the digital integrator in the previous example.

5.5 TRANSIENT RESPONSE ESTIMATE

In the direct matrix solution, response time is limited by the time required to obtain a good estimate of the covariance matrix and by the arithmetic unit used to compute the solution. For the steepest descent algorithm, the analysis of convergence time is more complicated.

The computation load for the digital realization of the loop is much lighter than for the direct matrix algorithm. There is basically one vector multiplication per iteration. As pointed out previously, the numerical accuracy requirement is relaxed, which also contributes to the speed of the multiplier. Analog circuits have been built and tested at instantaneous bandwidths of 150 MHz.

The problem lies in estimating how many iterations are required, or, equivalently, predicting the time constants of the control loops. To gain some insight into the problem, consider Equation 5-24 with V^*V^t replaced by its expected value, M.[†] It becomes

$$W(t) = S^* - h(t)\ (*)\ k\ M\ W(t) \tag{5-25}$$

This equation suggests a vector servo loop of the form shown in Figure 5-4(a). Since M contains nonzero off-diagonal terms, in general, the N loops implied by

[†] The implication of this assumption is that the noise process remains stationary over the transient response time of the processor.

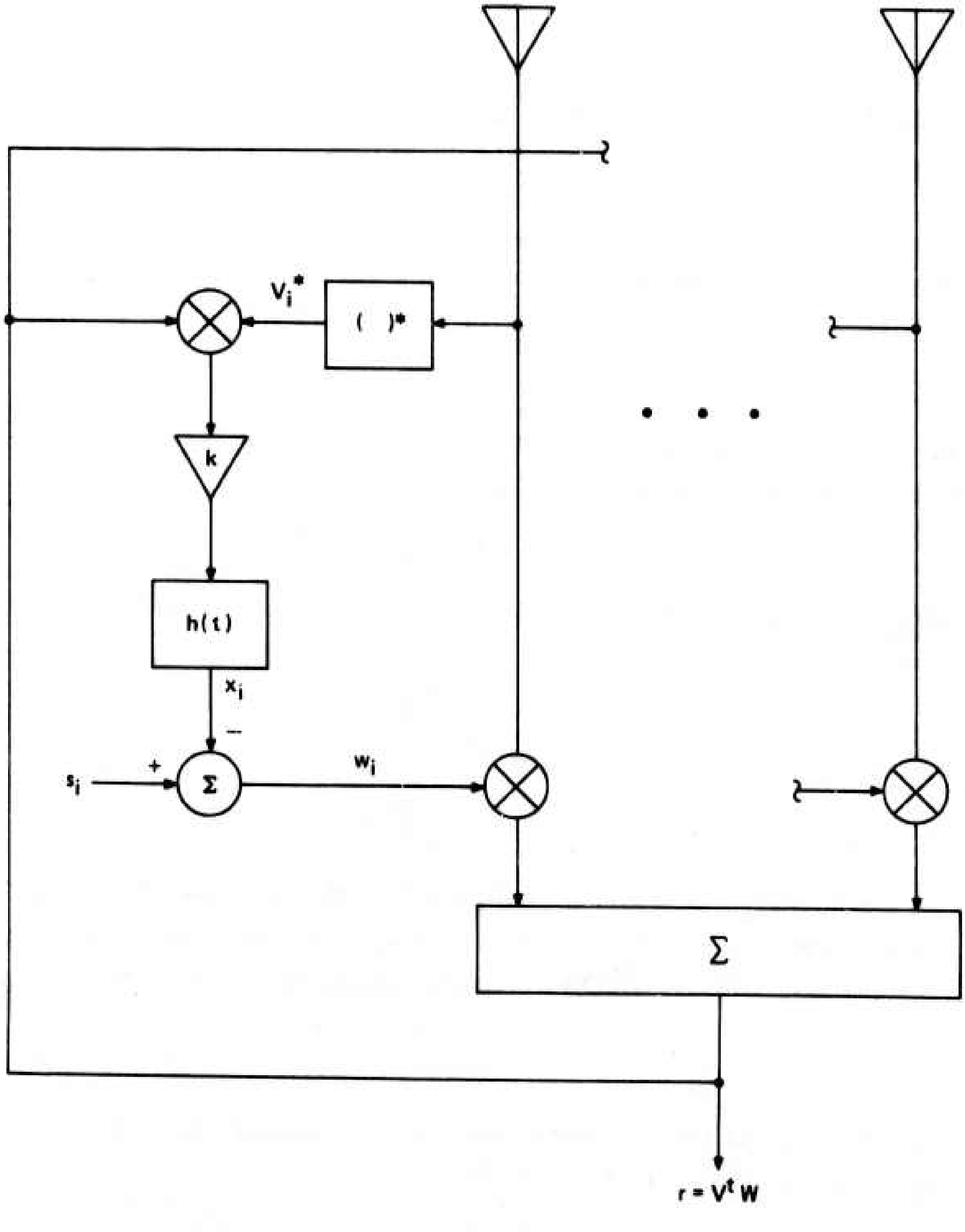


Figure 5-3. Analog (Howells) Co:trol Loop

the vector servo loop are cross-coupled. This makes analysis using conventional operational calculus techniques difficult.

It is always possible, however, to transform the problem to decoupled or "normal" space, thanks to the special properties of the matrix M. Consider a matrix A, with the following properties:

$$A^H M A = K \qquad (5\text{-}26)$$

where M is the usual covariance matrix and K is a diagonal matrix whose elements are, in fact, the eigenvalues of the matrix M. With no loss in generality we may also require, for convenience, that the matrix A be unitary. That is,

$$A^H A = I \qquad (5\text{-}27)$$

The columns of A are the eigenvectors corresponding to the eigenvalues in K. Consider the normal weights Z given by

$$Z = A^H W \quad \text{or} \quad W = A\ Z \qquad (5\text{-}28)$$

Substituting this relationship into Equation 5-25 and premultiplying by A^H yields

$$Z = Y = h(t)\ (*)\ K\ Z \qquad (5\text{-}29)$$

where

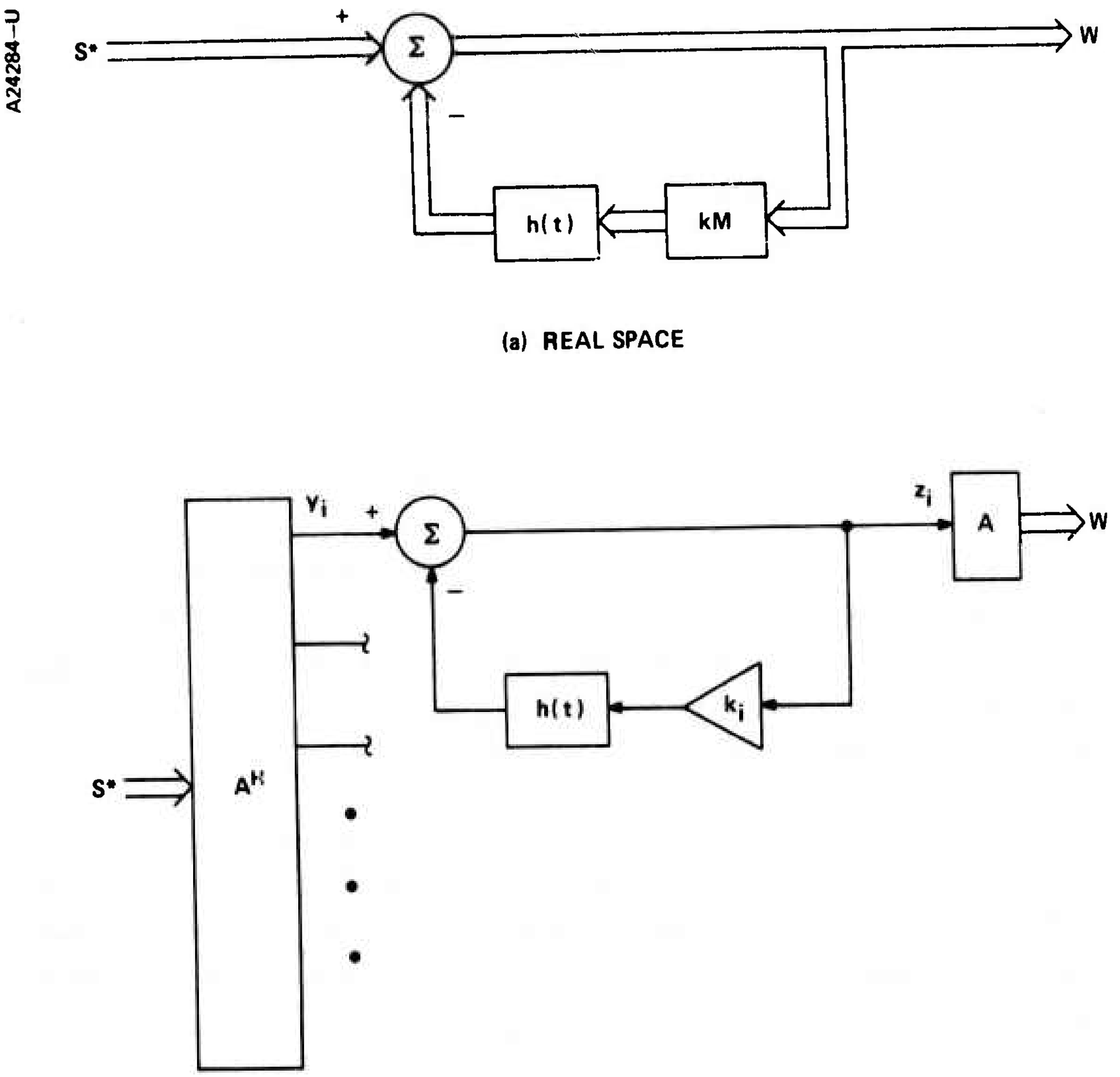
$$Y = A^H\ S^* \qquad (5\text{-}30)$$

This transformation is pictured in Figure 5-4(b) where only one of the now decoupled loops is shown. As previously mentioned, the filter, h(t), is most commonly a simple single-pole, low-pass filter of the form

$$H(s) = \frac{1}{1 + T_I s} \qquad (5\text{-}31)$$

where H(s) is the LaPlace transform of h(t). Consider the response of this system to a step input. It is of the form

$$z_i(t) = \frac{y_i}{1 + k_i}\left[1 + k_i\ \exp(-\ \frac{t(1 + k_1)}{T_I})\right] \qquad (5\text{-}32)$$



(a) REAL SPACE

(b) NORMAL SPACE

Figure 5-4. Vector Servo Loop

This expression is a simple exponential (as one would expect from the form of the filter) with enhanced time constant, T_{ei}, given by

$$T_{ei} = \frac{T_i}{1 + k_i} \tag{5-33}$$

In other words, each loop in normal space has a different time constant, a function of the eigenvalue, k_i, associated with that loop.

If the transformation to normal space is applied to Equation 5-15, the expression for the steady-state residue, the result is

$$P_r = \sum_{i=1}^{N} R_i \tag{5-34}$$

where

$$R_i = |y_i|^2 \frac{k_i}{(1 + k_i)^2} \tag{5-35}$$

Not all R_i are of significant magnitude in a given case. It is, therefore, necessary to examine the enhanced time constant, T_{ei}, for each residue fraction R_i which is of significant amplitude in order to estimate the transient response time. It will be noted that both T_{ei} and R_i are functions of the eigenvalues, k_i, and of the transformed desired signal, y_i.

The eigenvalues and the eigenvectors which comprise the matrix, A, are related to M which, in turn, is a function of the array geometry and the particular arrangement of the noise sources at a given instant. Thus, the transient response (or convergence time) of this algorithm is somewhat more dependent upon the statistics of the noise environment than the direct matrix algorithm.

Geometrically, the matrix A can be pictured as a coordinate transformation which aligns the coordinate system with the principal axes of the paraboloid described by Equation 5-15. The relative amplitudes of the eigenvalues describe the elipticity of the surface. A wide spread in amplitudes implies a high degree of elipticity. If initial conditions place one near one of the longer principal axes, the subsequent descent is not very steep and many steps are required to reach the optimum solution.

One gathers from the foregoing that, while the normal mode analysis (as it is termed) provides a good deal of insight into the "post-mortem" examination of particular numerical examples, it is still rather difficult to predict transient response in general, using this approach. Appendix G derives bounds for the time constants T_{ei}. The fastest time constant is limited to $10/(\pi B_s)$ by loop noise considerations, where B_s is the signal processing bandwidth. This corresponds to about three samples in the digital system if the sampling rate is $1/B_s$.

Theoretically, the slowest time constant is longer by a factor of $Q\ N(P_j/P_n)^{1/2}$ where Q is a circuit constant (usually about three) and the JNR is measured at one of the channel inputs to the processor. For a 100-loop system and a 40 dB JNR, this factor could be as much as 3×10^4. In various simulations, it has been observed that these slow modes are not usually important to the output residue. Over those modes with significant R_i factors, the variation is more on the order of N:1. This means that the longest important time constant in a digital system is about 3N samples.

For an exponentially decaying residue, the residue goes down by about 8 dB (1/e) for each time constant. It, therefore, takes about three time constants to obtain better than 20 dB cancellation. In the slower cases, this means that it would take about 9N samples to achieve at least 20 dB cancellation. In the most favorable case, it could take as few as 9 or 10 samples.

The variation in time constant with loop gain (or eigenvalue magnitude) is due to the shape of the low-pass filter, h(t). For a simple single-pole response, the function $/H(jw)/^2$ decreases at a constant rate of 20 dB per decade of frequency. Thus, the closed loop bandwidth varies approximately the same way. Using a higher order filter reduces the variation such that, with the same noise bandwidth, the speed of response can be increased by as much as 4:1. The average slope of the open-loop frequency response function is steeper resulting in less variation with loop gain over the dynamic range of the system.

5.6 COMPARISON OF ALGORITHMS

The direct matrix algorithm and the steepest descent algorithm can be compared in two important ways. The first is implementation complexity and the second is convergence rate or transient response time.

From previous discussions in this section we have concluded that the direct matrix algorithm requires a special-purpose processor, some type of vector arithmetic unit, for most real-time applications of interest. If this is the case, the computation time is proportional to N^2 (N is the number of input channels) and might take 6 ms for a N = 100 case. This would result in a set of weights which brought the residue within 3 dB of theoretical optimum.

The steepest descent algorithm would make use of the same vector arithmetic unit to perform the weight-and-sum operation. If it takes between 10 and 9N samples to achieve 20 dB cancellation (see previous page) then the equivalent computation time would be

$$10/B_s < T < 9N/B_s \qquad (5\text{-}36)$$

where B_s is the signal bandwidth and is also assumed to be the digital sampling rate. If the bandwidth were 10 MHz and there were 100 input channels, T would be between 1 and 90 μs.

In this attempt to compare the performance of the two algorithms, there is unavoidably an element of "apples versus oranges" since the method of analysis was very different for the two cases. What is needed is a statistical analysis of the steepest descent algorithm on the basis of signal-to-noise performance comparable to the one cited in Appendix F. Appendix H outlines one possible start in that direction.

Until such an analysis is done, one can only point out that, for real-time applications, the direct matrix algorithm is limited to those systems with a low N x B_s product primarily because of computation speed limitations. It has the advantage, however, of predictable convergence time.

SECTION 6

THE WEIGHT-STORAGE ADAPTIVE ARRAY

6.1 PROBLEM STATEMENT

In the study of the ballistic missile defense problem, it has been shown that precursor jamming poses a major threat to the proper operation of the radar sensors. A typical radar system for this application employs a phased array antenna with 3000 to 5000 elements per face. The precursor jamming threat consists of many small parachuteborne jammers distributed as uniformly as possible over the defense module and slowly descending to the ground.

One of the principal advantages of an electronically-steered phased array is that it can be rapidly scanned in a relatively short time to several diverse parts of the coverage volume in order to perform the various search and track functions needed. The most straightforward way in which to design an adaptive system to aid in nullifying the jamming is to reinitialize the adaptive processor each time the beam is switched, since the scanning of the array significantly changes the inputs to the processor. Thus, while the jamming environment is changing relatively slowly, the rapidly scanning antenna places a rather artifical requirement[†] on the transient response of the processor.

6.2 PROPOSED SOLUTION

It is possible to take advantage of the slowly moving jammer field in several ways in order to sidestep the scanning problem. Particularly for the search function, there are a finite number of discrete beam positions employed. If the proper adaptive weights for each beam position are stored in a memory and used each time the beam is switched to that position, then there will be no transient in the adaptive array output. The problem, of course, is to have the correct weights stored initially. The way in which this is accomplished depends upon the algorithm used.

[†] The adaptive processor should reach steady-state within approximately the first 100 μs after the beam is switched in order to provide performance against close-in targets.

For the direct matrix algorithm, the covariance matrix inverse is first accumulated. This quantity depends only upon the positions of the jammers and not upon the steering of the array. Since at least one constraint will usually be employed, the methods of Appendix D (see Equation D-6) are applied sequentially for each beam position to compute the weights. The new weights are then stored in memory in place of the previous estimate. Once the weights for each beam position are updated, the whole process is repeated. Meanwhile, the beam-steering processor uses the weights stored in memory as needed to scan the beam. The entire refreshing process must be repeated often enough to account for the motion of the jammers relative to the array.

For the loop-type processor, a slight variation of the same technique is used. Each time the beam is steered to a given position, the weights for that position are used as initial conditions in the loops. During the dwell at that beam position, fine adjustments are made to the weights according to the motion which has occurred since the last update. Just prior to scanning to a new beam position, the updated weights are stored back in the memory to become the initial conditions for the next time. When the system is first activated or when a new jammer first turns on, there will be a noticeable transient in the output residue but this will soon die out. Again, the restriction is that each beam position be visited often enough to keep up with the motion of the jammers.

Except for the possible future use of Charge-Coupled Device (CCD) memories, the weight storage is most easily accomplished digitally. If an analog system is desired, a hybrid loop approach can be employed in which an analog modulator is used for weighting the analog inputs, a single high speed analog-to-digital (A/D) converter is used in the residue path, and a digital multiplier is used in place of the analog correlator. The modulator could either be driven with digital weights directly, or a digital-to-analog (D/A) converter could be inserted for the purpose. Note that this technique avoids the use of individual A/D converters in each element input. Hard limiters (one bit A/D converters) are used to drive the digital correlators.

In any case, the crux of the entire weight storage question centers upon the relationship of the motion of the jammers to the update time of the processor. In order to obtain a feeling for the interaction between jammer motion and update time, a simulation study was undertaken.

6.3 SIMULATION - GEOMETRY

Figure 6-1 depicts the assumed geometry for the simulation. Jammers were assumed to be randomly distributed in a horizontal plane covering the area defended by the radar. The rate at which the jammers descended was made a variable parameter.

The array was assumed to consist of 5184 elements arranged in a rectangular aperture and on a halfwave square grid of rows and columns. The adaptive processor was configured in a row-column connection with a total of 144 degrees of freedom (72 rows and 72 columns).

6.4 SIMULATION - METHOD OF ANALYSIS

There are at least three ways in which the simulation could have been designed. If the voltages at the antenna due to jamming and thermal noise had been simulated by random number generators, then either the direct matrix algorithm or the digital form of the steepest descent algorithm could have been employed. The method which was chosen employs the smoothed differential equation describing the analog version of the control loop or steepest descent algorithm.

Using the known array and jammer geometry, covariance matrices were computed for 20 instants in time corresponding to "snapshots" taken once a second during a selected portion of the jammer cloud descent. Each matrix was then solved for its set of eigenvalues and the corresponding modal matrix.

For the first interval, the quiescent weights for the array were transformed to normal coordinates and used as initial conditions in the weight equations. These equations are of the form

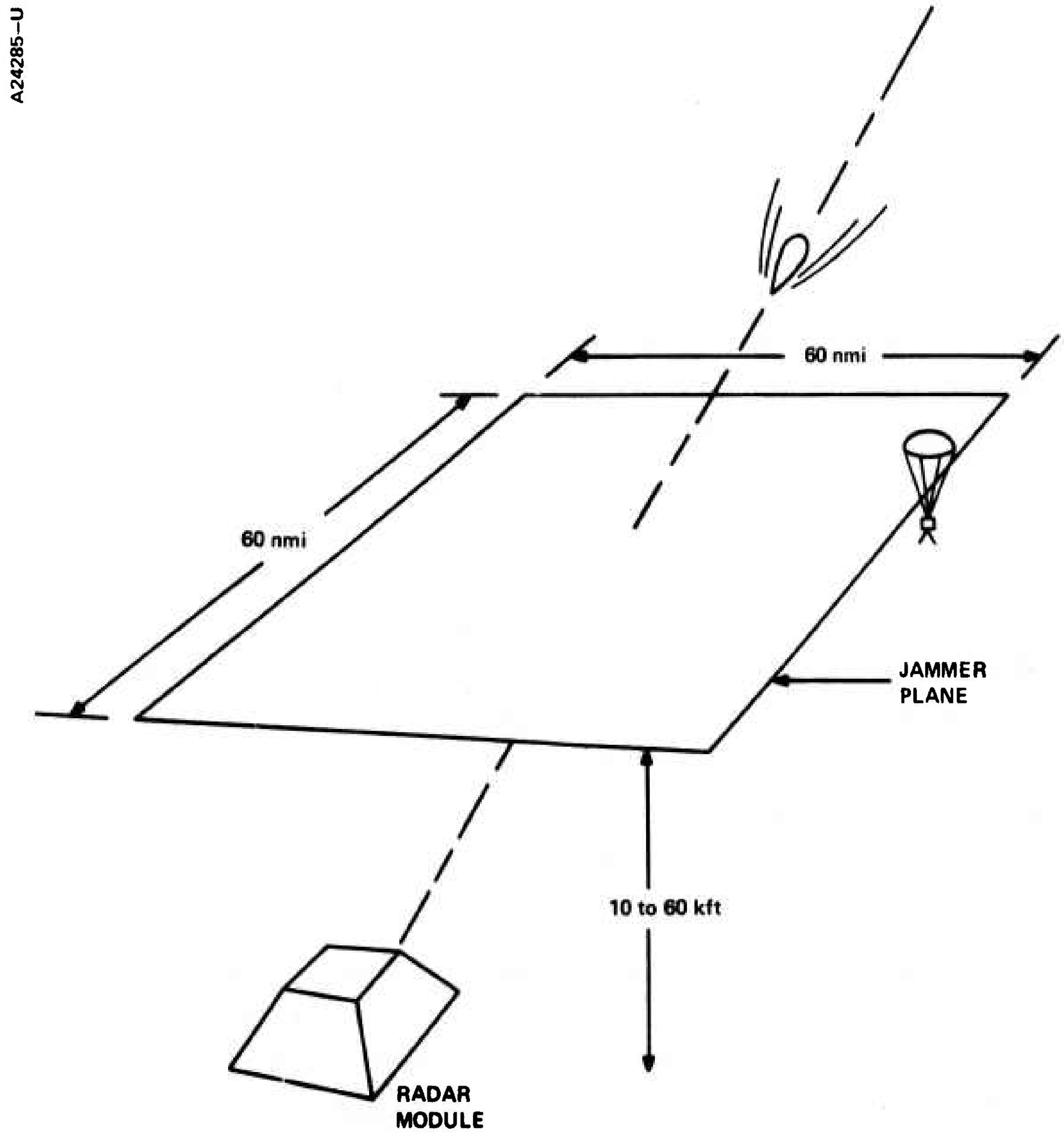


Figure 6-1. Precursor Jammer Geometry

$$z_i(t) = z_{oi} \exp(-t/T_{ei}) + z_{fi}(1 - \exp(-t/T_{ei})) \qquad (6\text{-}1)$$

where z_{oi} and z_{fi} are, respectively, the initial and final values of z_i and T_{ei} is the enhanced time constant for the i'th mode (see Section 5 and Appendix G for details on the transient analysis).

Time was then allowed to vary from zero to the end of one radar beam dwell time. For the simulations, one dwell was assumed to be about 200 μs. The normal mode weights for this time were then transformed back into real space and stored.

The entire process was then repeated for the second and succeeding intervals using the final values for the weights on one cycle as initial conditions in the next cycle. During each dwell period, the residue was computed and plotted.

Of the simulations run, two cases were selected for presentation as representing typical performance. Both cases consisted of a cloud of 200 jammers randomly distributed in a plane 60 x 60 nmi centered over the defended area. The nominal rate of descent was fixed at 1500 ft/min. In the first case, random array errors equivalent to -5 dBi were assumed. Total steady-state cancellation was about 14 dB (limited by the array errors). Figure 6-2 shows a plot of the transient response, that is, the level of the residue with respect to the final steady-state value. For each vertical line segment, the upper bound represents the initial value of the residue during that sweep while the lower mark shows the residue at the end of the sweep.

The upper curve shows the performance assuming a onc second frame time. Two and three second frame times are shown in the center and lower curves (alternatively, the middle and lower curves could represent descent rates of 3000 and 4500 ft/min at a 1s frame rate).

Note that steady-state is all but reached in one sweep. Motion of the jammers between looks causes a slight regression in the residue, which is somewhat worse for the longer frame times.

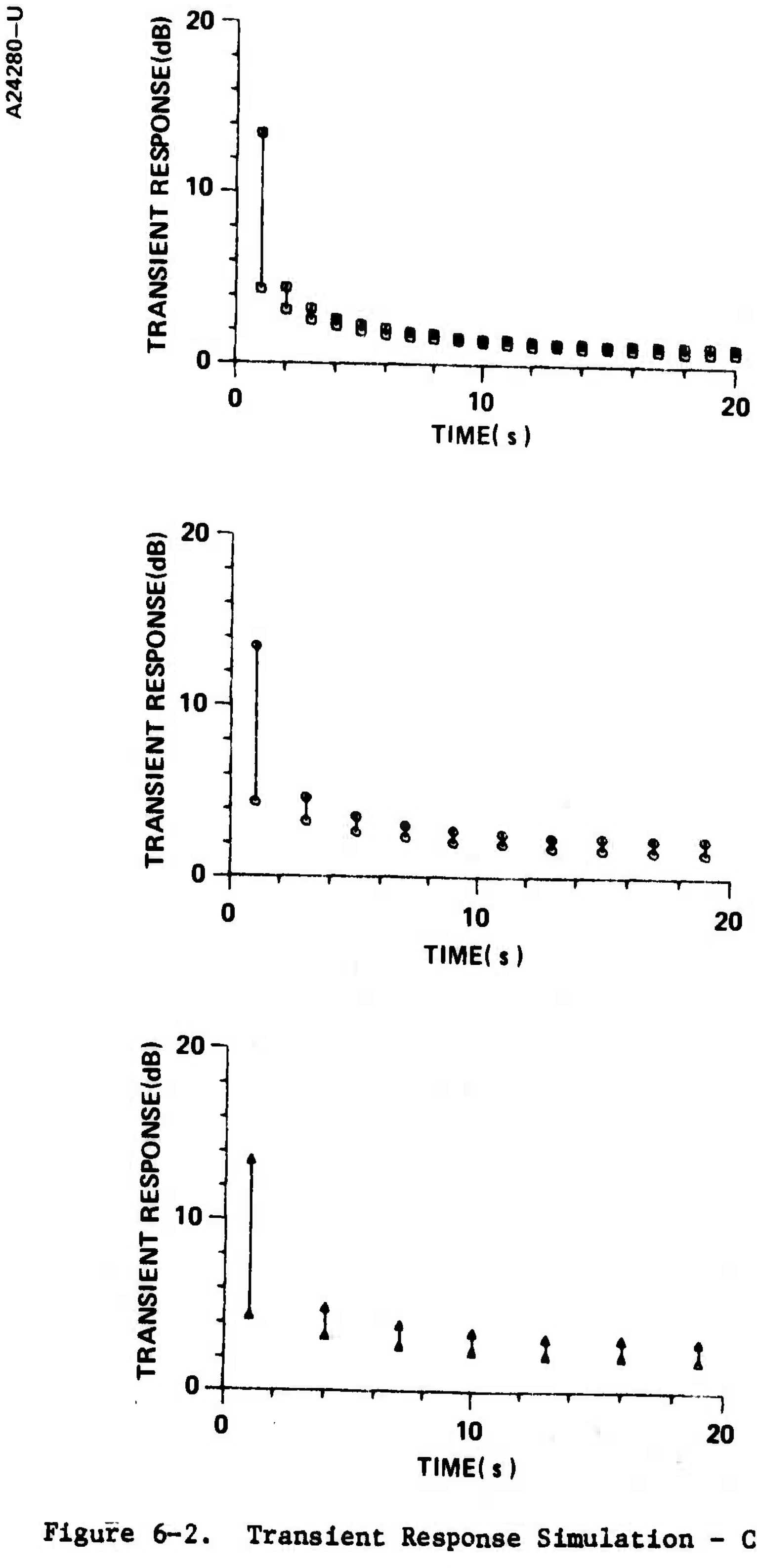


Figure 6-2. Transient Response Simulation - Case 1

In order to determine the effect of errors on the transient performance the error level was lowered considerably in the second case, shown in Figure 6-3. The steady-state cancellation was on the order of 38 dB (note that the vertical axis extends only to 20 dB). Over 25 dB of cancellation is achieved in the first sweep. Sweep-to-sweep regression is higher in this example because the absolute magnitude of the residue is considerably smaller than in the first example.

The results of these simulations indicate that the precursor jammer threat exhibits sufficient stationarity to allow the weight storage technique to be effective in overcoming the transient response limitation caused by rapid array scanning requirements.

For this simulation, simple single-pole filters were assumed for the steepest-descent loop equations. If higher order filters were employed, the response could have been speeded up by a factor of 3 to 4:1.

It must be pointed out that the weight storage technique depends upon maintaining the stationarity of the jammer environment. This has implications concerning the use of frequency agility. A separate set of weights must be stored for each beam for each radar frequency channel used. For the loop algorithm, this poses no particular problem since each beam is essentially an independent calculation. All that is needed is to have a particular beam always used at the same frequency. As the system is scanned from beam to beam, a sidelobe jammer sees an apparently frequency-agile radar since subsequent beam positions would use different frequency channels. If more than one frequency is required for a particular beam position, additional receiver channels would have to be supplied and parallel computations performed for each frequency.

The direct matrix algorithm computes one matrix for all beams for one frequency. It would, therefore, have to repeat all calculations for each new frequency for all beams used in the scan. One would then, however, have the freedom to use any frequency at any beam position.

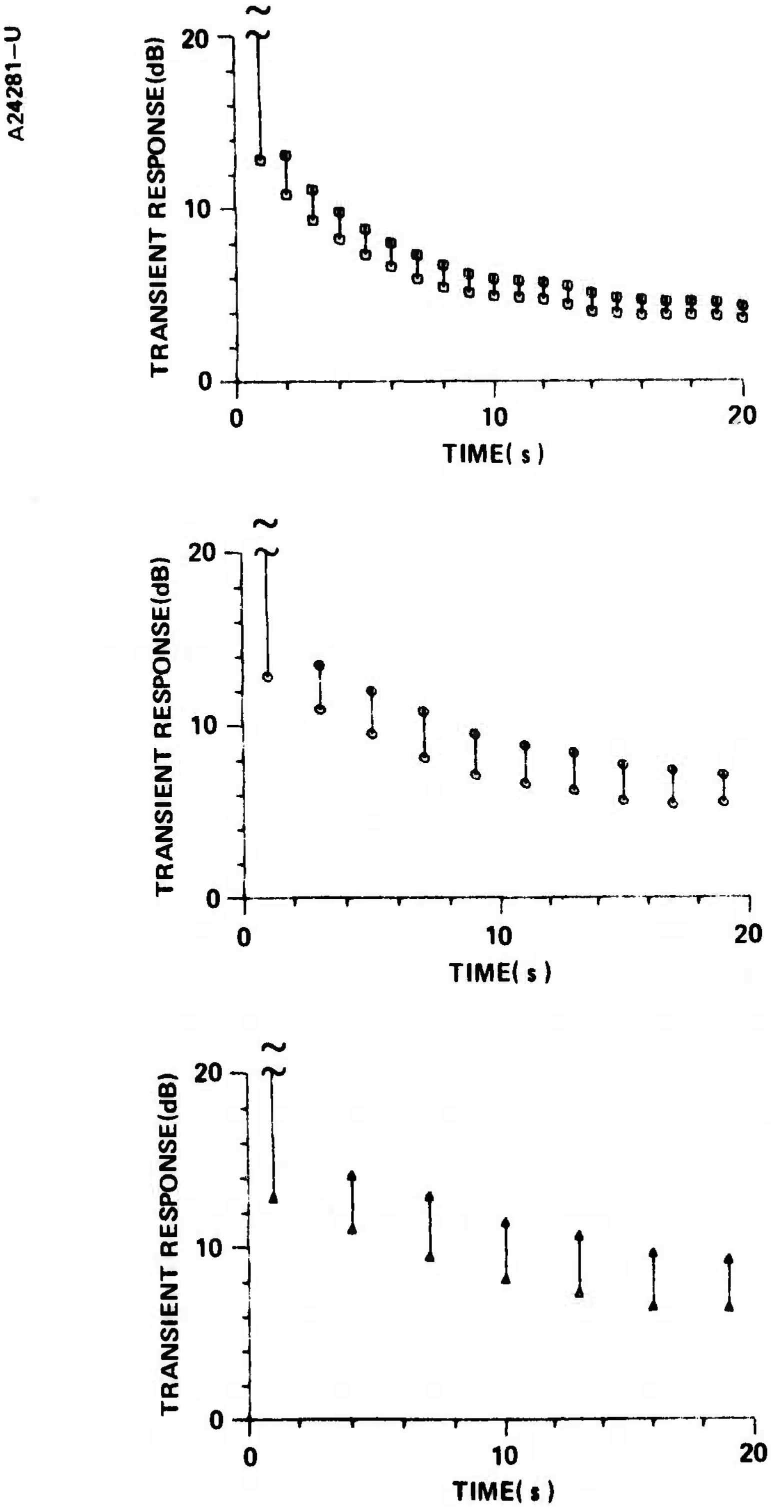


Figure 6-3. Transient Response Simulation - Case 2

SECTION 7

DISCRETE BEAM TRACKING STUDY

7.1 INTRODUCTION

The feasibility of acquiring and tracking reentry vehicles (RVs), using only those radar beam positions used in search mode, has been investigated. The results obtained do not show that the difficulties with discrete beam tracking can be easily overcome by a nominal increase in transmitted energy, as had been conjectured. The primary difficulty with the discrete beam tracking technique is that, early in a track, when the error ellipse is large, selection of the optimum beam for track pulse transmission is an uncertain operation. In a low signal-to-jammer ratio (SJR) environment, wrong beams are frequently selected, and the resulting high beam-shape loss causes missed detections or, at best, increased measurement errors. This effect leads to a greater number of precommit track losses, and also makes the impact point prediction and intercept planning tasks more uncertain, which is especially important in a low-altitude ballistic missile defense (BMD) system.

The problems of discrete beam tracking can be countered by a more careful track initiation procedure than was used in obtaining the present results, as well as a local search procedure which would facilitate recovery from a wrong beam selection. Another effective step would be to decrease the packing factor of the search beam lattice. The beam-shape loss caused by wrong beam selection would then be decreased.

7.2 METHOD OF GENERATING RESULTS

The Advanced ECCM* Radar Simulation(10) developed for the Ballistic Missile Defense Advanced Technology Center (BMDATC) under an earlier contract was used to simulate an attack of 60 RVs on the 8 silos closest to the radar, as shown in Figure B-2. Trajectory elevation angles were evenly distributed between 15°, 25°, 35°, and 45°. Trajectory azimuths were randomly distributed over

*Electronic Counter-Countermeasures

the range from -20° to +20°. Track initiation was attempted at ranges corresponding to the far boundary of a search volume having a 60 to 80 kft altitude horizontal section, tapered at the lower elevation angles. The average acquisition range was 154 kft. The beam-center SJR at acquisition (SJR_o) was varied (12, 15, 18, 21, 23 dB) for each simulation run, as was the beam steering method (discrete or continuous). Gaussian measurement errors were simulated.

Reentry vehicles were assumed to be detected and verified by the search beam closest to the true RV position. Track initiation pulses were transmitted until the one-dimensional angle rate error variance was reduced to 0.05 rad^2/s^2, or an acquisition failure occurred. For RVs which were successfully acquired, Kalman filter tracking at a 20 Hz rate for 6 s was attempted. Statistics were compiled on acquisition failures, track losses, and position tracking errors.

For the case of continuous beam tracking, the radar beam was steered directly at the predicted direction cosine estimates provided by the polynomial or Kalman filter. For discrete beam tracking, the beam was steered at the search beam position closest to the predicted estimates. The search volume packing factor was 0.88.

7.3 RESULTS AND CONCLUSIONS

Percentages for successful acquisition and tracking are given in Figure 7-1. For the cases of very high or very low SJR_o, the success percentages of the two beam steering techniques are seen to be essentially equivalent. For the middle range of SJR_o, substantial increases in transmitted energy are seen to be needed for the discrete system to match the performance of the continuous system. For a 95% probability of acquisition, for example, the discrete system needs about 4.5 dB higher SJR_o than the continuous. For a 90% probability of successful tracking, 6.5 dB higher SJR_o is required.

Tracking accuracy statistics, shown in Figure 7-2, also indicate that more than a nominal 3 dB energy increase is needed for the discrete beam system to match the continuous.

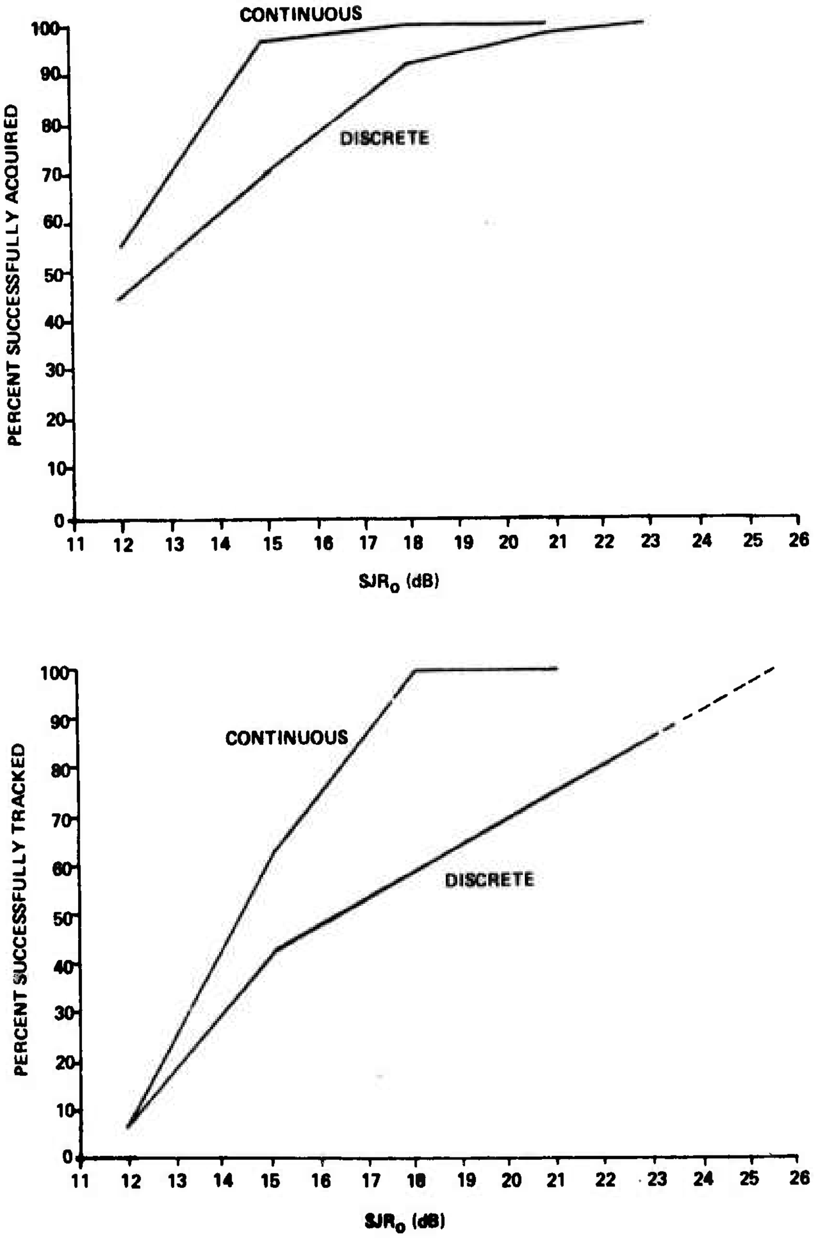


Figure 7-1. Acquisition and Tracking Performance Comparisons

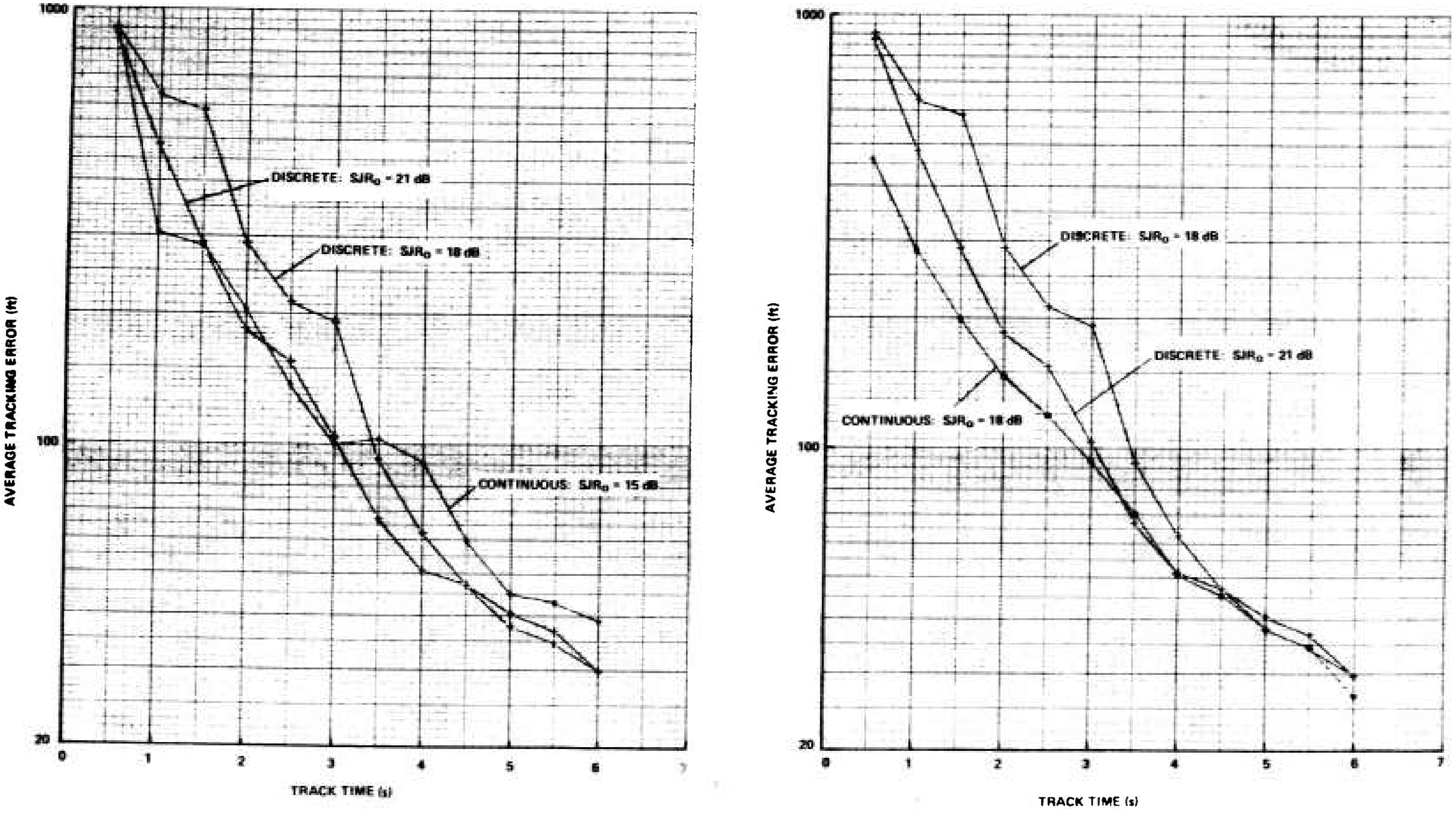


Figure 7-2. Tracking Accuracy Comparison

Figure 7-3 illustrates the primary difficulty with discrete beam steering - wrong beam selection. The track at the top, initiated with a relatively high SJR, experiences little difficulty in choosing the closest search beam. At the bottom of Figure 7-3, the same trajectory is flown, but the lower SJR causes a wrong beam to be chosen early in the track. When this occurs, gaps in the data result, and the track is frequently lost. At best, a serious degradation in tracking performance occurs in the crucial precommit track period.

The three methods suggested earlier for overcoming the difficulties of discrete beam tracking are:

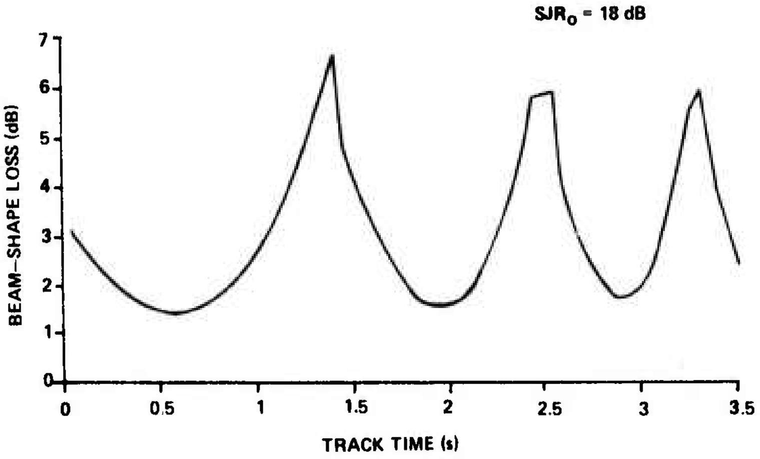
1. A better track initiation procedure. The procedure used in generating the current results is one which has been found to work well when continuous beam steering is used. For discrete beam steering, a useful modification would be to use a single, non-standard beam* for initiating each track. The gain in acquisition probability associated with this procedure would probably far outweigh the price which must be paid in settling time. The same non-standard beam, centered on the estimated target position, would then be used for all track initiation pulses.

2. A more active local search function. When continuous-beam steering is used, a tracking system will normally enter a local search mode after a fixed number of consecutive missed hits. If discrete beam steering is being used, the local search "threshold" needs to be set lower, because of the possibility of wrong beam selection. The Advanced ECCM Radar Simulation at present lacks the local search model needed to properly evaluate the tracking performance of discrete beam steering.

3. A lower search beam lattice packing factor. Many of the discrete beam acquisition failures occurred for those targets which were near search beam boundaries. These targets suffered high beam-shape losses even when the correct beam was selected. An obvious way to counter this problem is to lower the packing factor; that is, to pack the search beams in more closely. Some of the time saved by weight holding may need to be expended in this manner.

Although these methods have not been actually evaluated in the simulation program, it is clear that their use will allow discrete beam tracking performance to approach that of continuous beam tracking as closely as desired.

*Not necessarily a search beam.



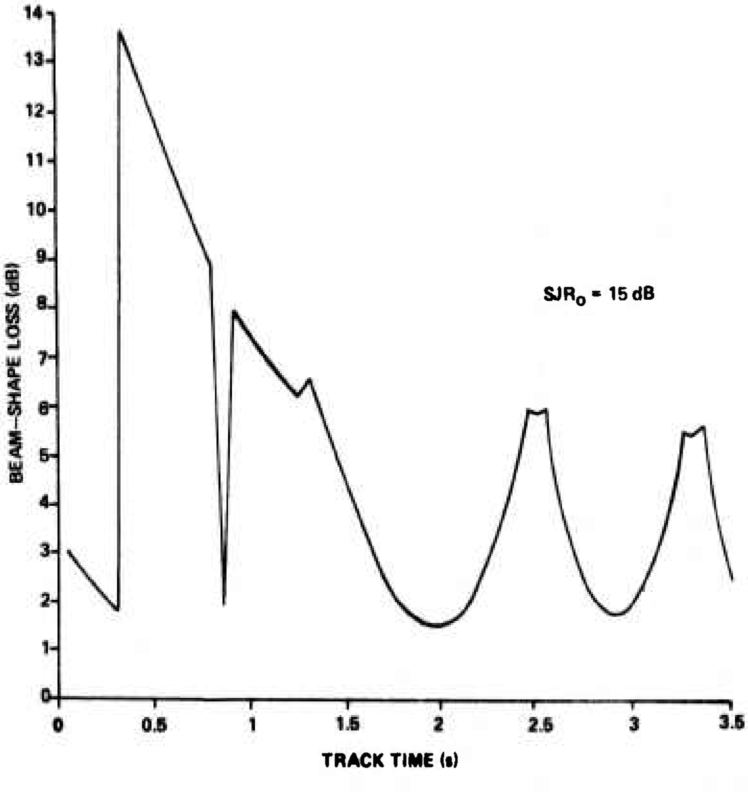


Figure 7-3. Beam-Shape Loss Profiles

GLOSSARY

A/D	Analog-to-Digital
BMD	Ballistic Missile Defense
BMDATC	Ballistic Missile Defense Advanced Technology Center
BW	Beamwidth
CCD	Charge-Coupled Device
D/A	Digital-to-Analog
ECM	Electronic Countermeasures
ECCM	Electronic Counter-Countermeasures
ICBM	Intercontinental Ballistic Missile
JNR	Jammer-to-Noise Ratio
LMS	Least Mean Square
MSLC	Multiple Sidelobe Cancellation
PF	Packing Factor
RCPAA	Row-Column Precision Adaptive Array
RCS	Radar Cross Section
RV	Reentry Vehicle
SJR	Signal-to-Jammer Ratio
SJR_o	Beam-Center SJR at Acquisition
SNR	Signal-to-Noise Ratio

REFERENCES

1. D.J. Chapman, ECM/ECCM Studies, Volume 3: "Adaptive Array Technique Study", Final Report prepared by the Syracuse University Research Corporation (SURC) for the U.S. Army Advanced Ballistic Missile Defense Agency under Contract DAHC60-73-C-0044, SURC TR 73-186, February 1974 (Unclassified).

2. S. Applebaum, J. Len, and D.J. Chapman, ECM/ECCM Evaluation Program, Volume 5: "Adaptive Array (U)," Final Report prepared by the Syracuse University Research Corporation for the U.S. Air Force Rome Air Development Center under Contract F30602-72-C-0075, RADC TR 73-340 (SURC TR 73-152), (Confidential).

3. J. Butler, "Multiple Beam Antenna," Sanders Associates Internal Memo, RF 3849, 8 January 1960.

4. P.W. Howells, "High Quality Array Beamforming with a Combination of Precision and Adaptivity," prepared by Syracuse University Research Corporation, SURC TN 74-150, June 13, 1974 (Unclassified).

5. S. Applebaum, "Adaptive Arrays," Special Report prepared by the Syracuse University Research Corporation for the Advanced Research Projects Agency under Contract AF30(602)-3523, SURC TR 66-001, August 1966 (Unclassified).

6. I.S. Reed, et al., "Rapid Convergence Rate in Adaptive Arrays," IEEE Transactions on Aerospace and Electronic Systems, Vol. AES-10, No. 6, p. 853.

7. S. Applebaum and D.J. Chapman, "Adaptive Arrays with Mainbeam Constraints," NRL Report #7803, Proceedings Adaptive Antenna Systems Workshop, Volume 1, September 1974 (Unclassified).

8. B. Widrow, et al., "Adaptive Antenna Systems," Proceedings of the IEEE, Vol. 55, pp. 2143-2158.

9. O.L. Frost, "An Algorithm for Linearly Constrained Adaptive Array Processing," Proceedings of the IEEE, Vol. 60, No. 8, August 1972.

10. W.J. Cole, R.M. Davis, and S. Kamak, ECM/ECCM Studies, Volume 5: "Advanced ECCM Radar Study (U)," Final Report prepared by the Syracuse University Research Corporation for the U.S. Army Advanced Ballistic Missile Defense Agency under Contract DAHC60-73-C-0044, SURC TR 73-186, August 1974 (Secret).

APPENDIX A

COMPARISON OF TRACKING PERFORMANCE WHEN DISCRETE-BEAM STEERING IS USED AND MEASUREMENT ERRORS HAVE EITHER GAUSSIAN OR PHASE COMPARISON DISTRIBUTION

One difficulty with discrete beam tracking was expected to arise from the fact that a phase comparison monopulse system yields biased measurements when the tracked object is off the nose of the beam(10). An attempt was made to determine the effect of this bias on discrete beam tracking accuracy. Surprisingly, an improvement, rather than a degradation, in tracking performance was observed when the simulated measurement errors were changed from gaussian to phase comparison. The improvement was most likely due to a calibration method used, as described in this appendix.

For the case of gaussian angle measurement errors, the standard deviation is governed (approximately) by the formula

$$\sigma = \frac{BW}{k_m \sqrt{SJR}} \tag{A-1}$$

Since the correct value to use for k_m has been controversial, it was determined to perform some calibration runs for various values of k_m and to pick the one which most closely approximated the phase comparison case. Twenty incoming RVs were simulated, all flying the same trajectory -- down the radar's mechanical boresight. Average tracking errors were calculated. At all times the radar beam was steered directly at the RV's true position -- not at its estimated position. The acquisition SJR was 21 dB. The results, shown in Figure A-1, indicate that $k_m = 1$ is an acceptable value. It was used for all the gaussian simulation runs.

Figure A-2 shows a comparison of tracking performance when discrete beam tracking is used with gaussian angle measurement errors, and with phase comparison angle measurement errors. Runs were made for $SJR_o = 12$ and 21 dB. Contrary to expectation, the tracking performance with phase comparison errors was significantly better than with gaussian errors. This occurred for $SJR_o =$ 21 dB as well as 12 dB.

Obviously, the effect of any measurement bias on discrete beam tracking accuracy cannot be analyzed unless or until a different method is found for determining equivalent-variance gaussian and phase comparison measurement errors.

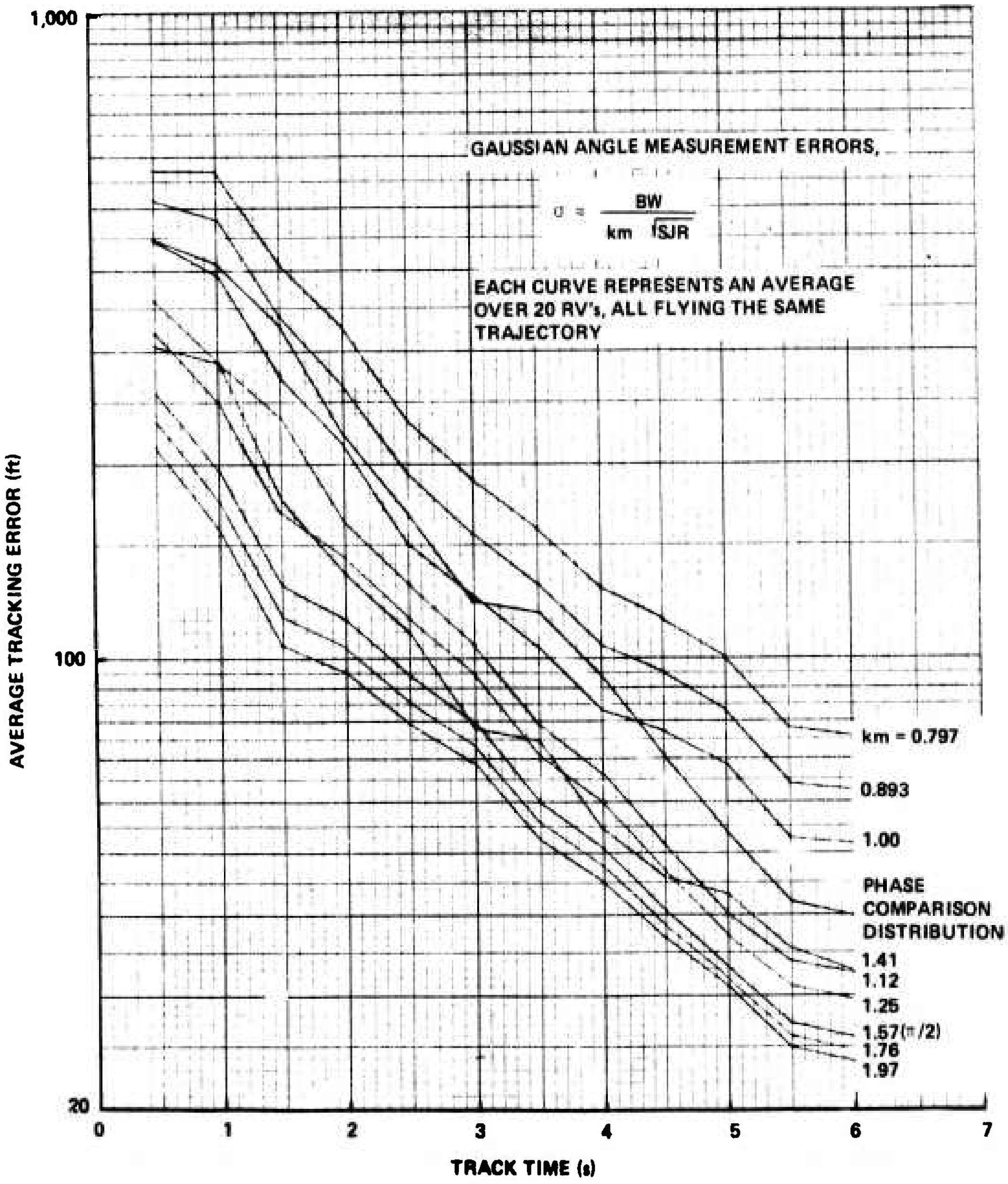


Figure A-1. Results of Calibration Runs

	ACQUISITION PERFORMANCE		TRACKING PERFORMANCE	
	21 dB	12 dB	21 dB	12 dB
GAUSSIAN	98%	45%	76%	7%
PHASE COMPARISON	100%	62%	90%	25%

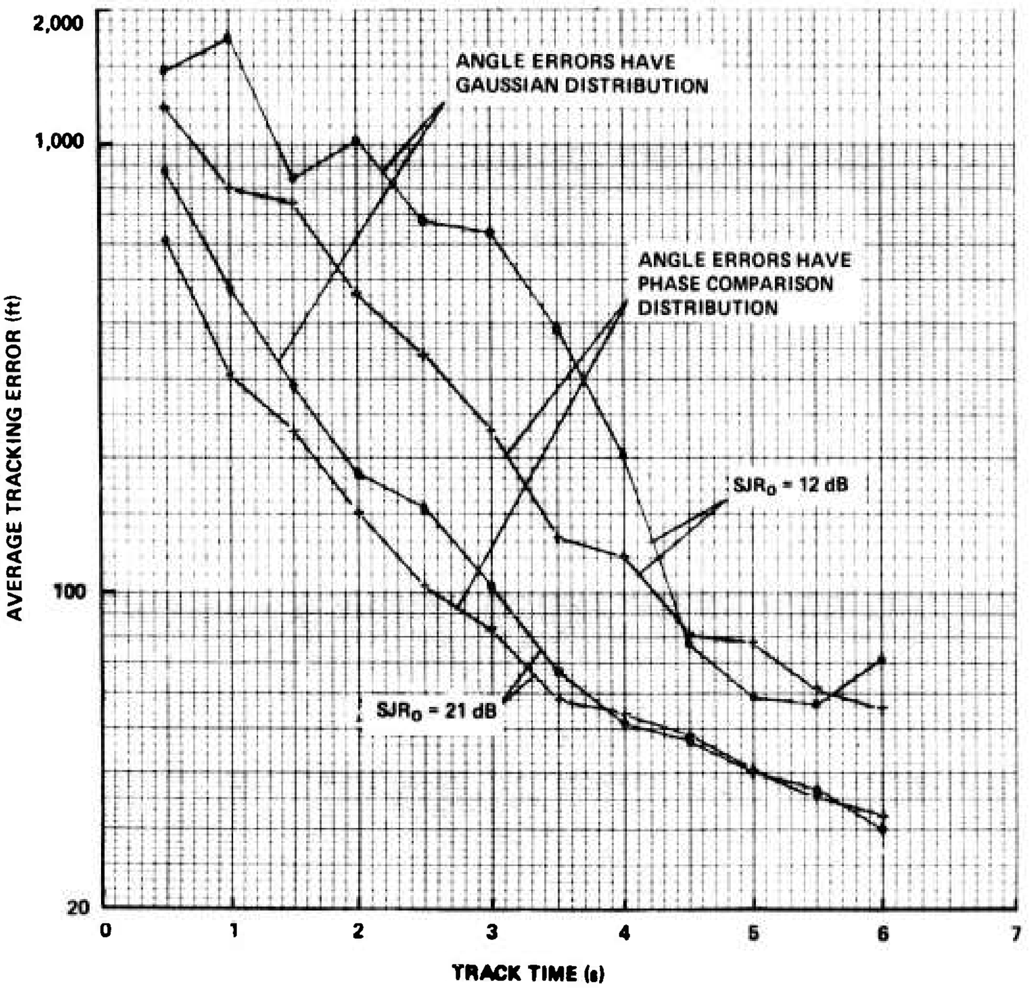


Figure A-2. Tracking Accuracy Comparison

APPENDIX B

TRAJECTORY STUDY IN COSINE SPACE

In the simulation runs made while studying discrete-beam tracking, the possibility of an RV flying outside the search volume was discounted. Instead, the entire sky was assumed to be filled by a lattice of search beams. This was justified on the grounds that if discrete beams were "strung" along a trajectory (Figure B-1) outside the search volume, the resulting tracking performance would closely approximate that obtained by assuming that RV trajectories would always be covered by a lattice of search beams.

In order to estimate the frequency of RVs flying outside the search volume during the course of an intercept, some worst-case trajectories were generated and plotted in cosine space. The trajectories were started at radar ranges where they might normally be detected and flown toward their target silos until they either passed out of the quadrant covered by the front face of the radar, or descended to an altitude of 5 kft.

The trajectories were worst-case in the sense that their elevations and azimuths were all on the extremes of the assumed threat distributions (i.e., 15° and 45° for elevation, and ± 25° for azimuth). Other trajectories, which passed directly over the tracking radar, were also flown and plotted. Target silos were the 16 closest ones to the tracking radar, as shown in the module layout diagram, Figure B-2.

The trajectory plots are shown in Figure B-3. The triangular contour in these figures represents the cosine space boundaries of the quadrant which would be assigned to the radar's north face, assuming the face is tilted 30° from the vertical. The search volume shown is one that might be assigned to the east radar if the two northernmost radars were assigned the task of Intercontinental ballistic missile (ICBM) search. The azimuth limits are -25° to +50°, and the elevation limits are 5° to 55°. The trajectories which are represented only as dots never entered the north quadrant and so were terminated immediately.

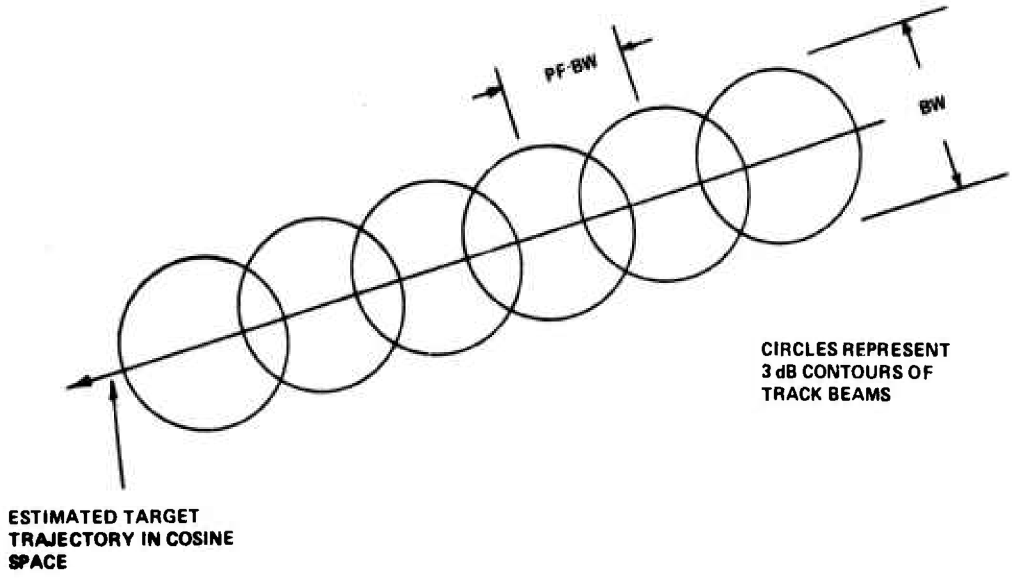


Figure B-1. "Stringing" of Beams Along a Trajectory

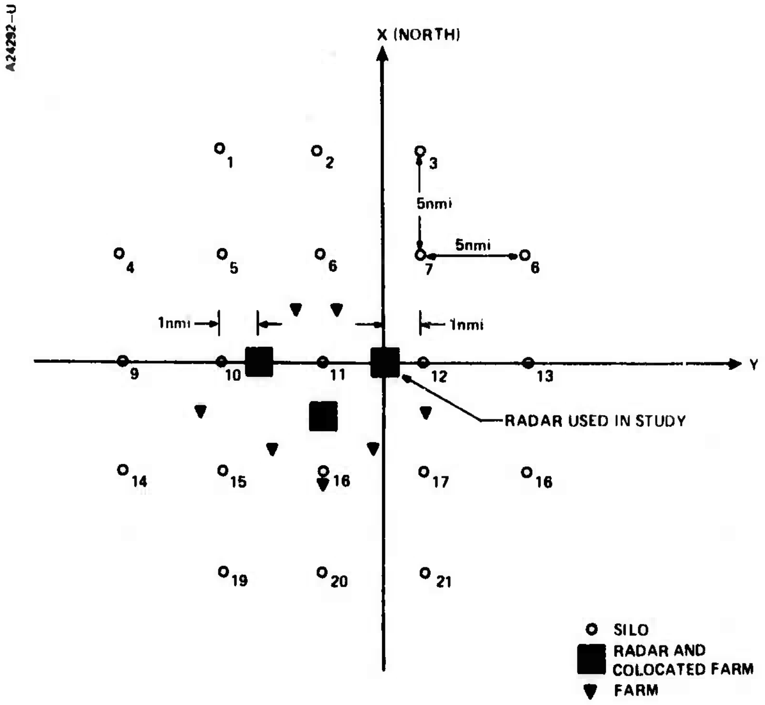
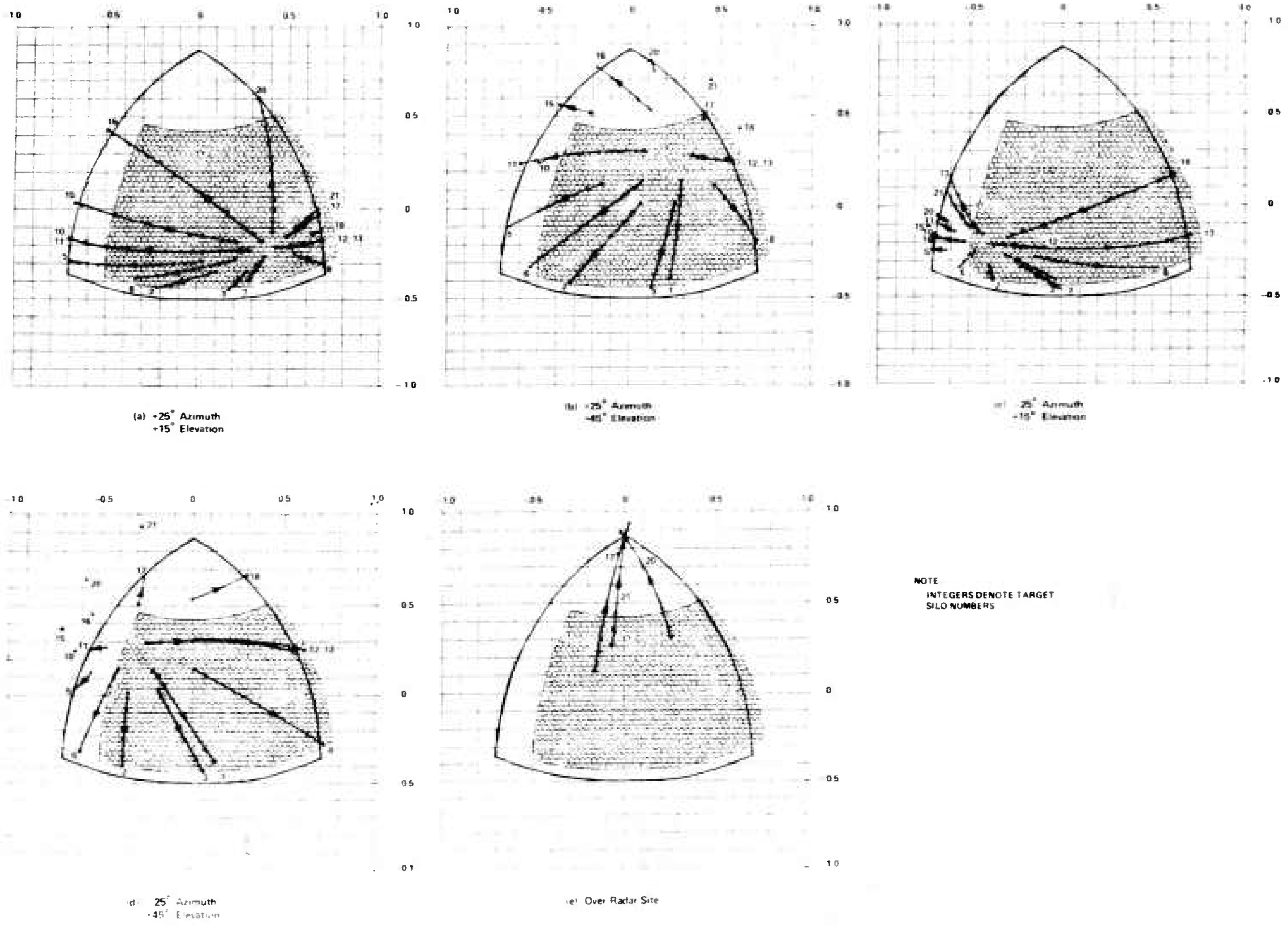


Figure B-2. Simulated Module Layout



(a) +25° Azimuth
+15° Elevation

(b) +25° Azimuth
-45° Elevation

(c) -25° Azimuth
+15° Elevation

(d) 25° Azimuth
-45° Elevation

(e) Over Radar Site

NOTE
INTEGERS DENOTE TARGET
SILO NUMBERS

Figure B-3. Cosine Space Representation of Trajectories at Various Azimuth and Elevation Angles with Respect to Target Silo

The primary conclusion, not unexpected, to be drawn from these plots is that only a small percentage (less than 10%) of the trajectory lengths are outside the search volume. Furthermore, the ones that are outside the search volume could be easily handled by the east face of the west radar in the module. The radar cross section (RCS) enhancement due to a more favorable aspect angle would most likely result in the tracking function for such trajectories being assigned to the west radar even without regard to canceller settling time. Discrete beam tracking, therefore, is found to be a workable procedure.

Another fact worth mentioning is that, near intercept, the typical trajectory is cutting across beams at a much more rapid rate than it was at detection. The time gained in stringing discrete beams along a trajectory outside the search volume is not likely to be substantial if the RV is changing beams every 0.2 s.

APPENDIX C

SIMULATION OF THE SEARCH FUNCTION AND DISCRETE BEAM TRACKING

Beam positions in a search volume in cosine space are arranged on a hexagonal closest-packed lattice (Figure C-1). The packing factor, denoted PF, is defined as the distance between beam centers divided by the 3 dB beamwidth.

The one-way beam-shape loss in dB, denoted $BSL_1(dB)$, associated with a given target position and a given beam position, is given by

$$BSL_1(dB) = 12\left(\frac{\theta_e}{BW}\right)^2 \qquad (C\text{-}1)$$

where θ_e is the angle between the nose of the beam and the target, and BW is the 3 dB beamwidth. This formula is an approximation which is accurate for $\theta_e \leq 0.9$ BW.

The worst-case $BSL_1(dB)$ associated with a search volume is related to the packing factor by

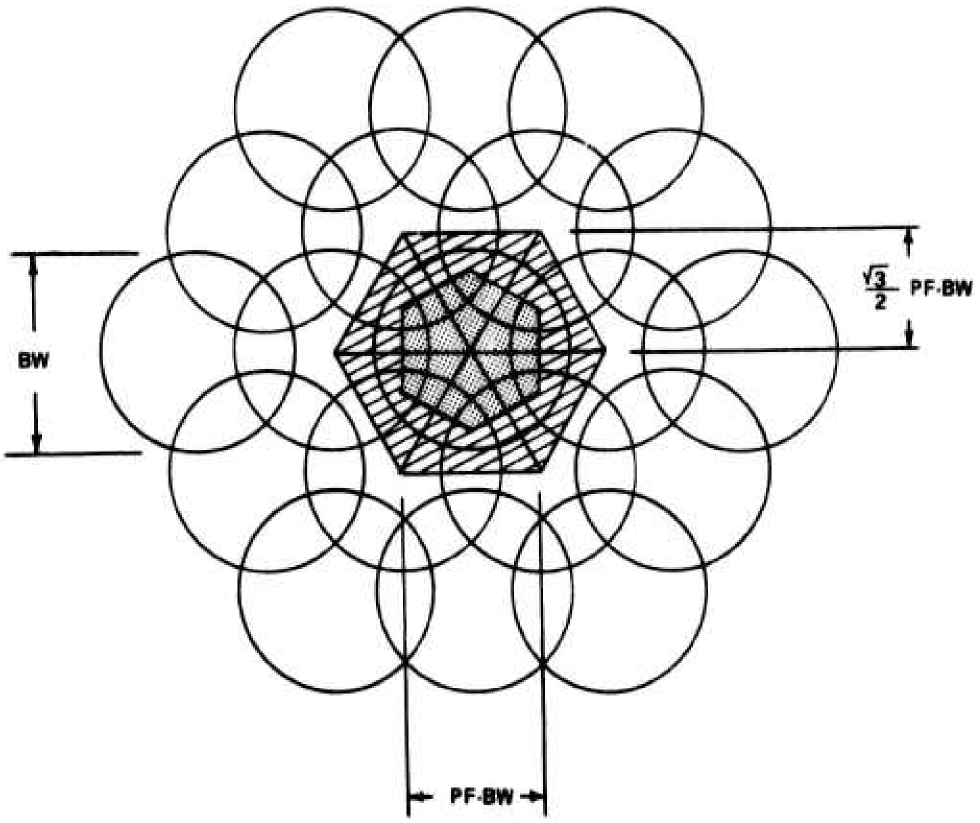
$$\text{worst-case } BSL_1(dB) = 4\ PF^2 \qquad (C\text{-}2)$$

When the worst-case $BSL_1(dB)$ is specified, and the required PF is computed, the following results are obtained:

Worst-case $BSL_1(dB)$	Required PF
3	0.866
2	0.707
1	0.500

When discrete beam tracking is being used, the center beam of the cluster shown in Figure C-1 would be selected only if the estimated RV position was inside the small hexagon shown in the figure. Each beam position has such a hexagon associated with it. The hexagon, therefore, represents the "effective area" of its beam, and it has an area of

$$A_e = \frac{\sqrt{3}}{2} \cdot PF^2 \cdot BW^2 \qquad (C\text{-}3)$$



NOTES:

1. PF = 0.707
2. EACH CIRCLE REPRESENTS 3 dB CONTOUR OF A BEAM
3. SMALL HEXAGON REPRESENTS EFFECTIVE AREA OF A BEAM:

$\frac{\sqrt{3}}{2} \cdot PF^2 \cdot BW^2$

Figure C-1. Search Beams on Hexagonal Closest-Packed Lattice in Cosine Space

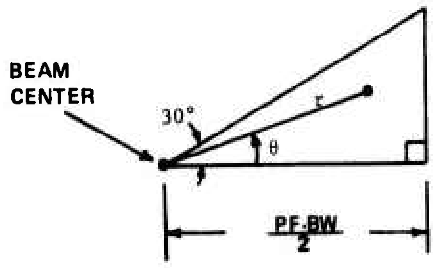
This relation can give the number of beams required for a search volume, if the cosine space area of the search volume is known. It also says that reducing the packing factor from 0.866 to 0.707 requires 1.5 times as many beams, and reducing it from 0.866 to 0.5 requires 3 times as many beams.

The average two-way beam-shape loss which would be experienced in discrete-beam tracking if correct beam selection always occurred is

$$\overline{BSL_2(dB)} = \frac{10}{3} PF^2 \qquad (C-4)$$

The assumption of correct beam selection has been found to be not valid early in a track, but for later in the track, this formula can be used to estimate the power increase needed to offset the discrete beam tracking loss. A derivation follows.

The small hexagon in Figure C-1 can be broken up into 12 similar triangles, one of which is shown in Figure C-2. The average two-way beam-shape loss will be computed over this triangle.



TWO-WAY BEAM-SHAPE LOSS
ASSOCIATED WITH A POINT =

$$BSL_2(dB) = 24\left(\frac{r}{BW}\right)^2$$

Figure C-2. Average Two-Way Beam-Shape Loss

The average two-way beam-shape loss is

$$\overline{BSL_2(dB)} = \frac{1}{A} \int_0^{\pi/6} \int_0^{\frac{PF\cdot BW}{2\cos\theta}} 24 \left(\frac{r}{BW}\right)^2 rdrd\theta \qquad (C-5)$$

where A is the area of the triangle.

$$\text{Let } u = \left(\frac{r}{BW}\right)^2;\ du = \frac{2rdr}{BW^2}\ ;\ rdr = \frac{BW^2}{2}\ du$$

$$\overline{BSL_2(dB)} = \frac{1}{A} \int_0^{\pi/6} \int_0^{\frac{PF^2}{4\cos^2\theta}} 12\ BW^2\ udud\theta = \frac{12\ BW^2}{A} \int_0^{\pi/6} \left.\frac{u^2}{2}\right|_0^{\frac{PF^2}{4\cos^2\theta}} d\theta$$

$$= \frac{12\ BW^2PF^4}{32A} \int_0^{\pi/6} \frac{d\theta}{\cos^4\theta} = \frac{3BW^2PF^4}{8A} \left[\frac{1}{3}\frac{\sin\theta}{\cos^3\theta} + \frac{2}{3}\tan\theta\right]_0^{\pi/6}$$

$$= \frac{5BW^2PF^4}{12\sqrt{3}\ A} \qquad (C-6)$$

But

$$A = \int_0^{\pi/6} \int_0^{\frac{PF\cdot BW}{2\cos\theta}} rdrd\theta = \int_0^{\pi/6} \left.\frac{r^2}{2}\right|_0^{\frac{PF\cdot BW}{2\cos\theta}} d\theta = \frac{PF^2BW^2}{8} \int_0^{\pi/6} \frac{d\theta}{\cos^2\theta}$$

$$A = \frac{PF^2BW^2}{8} \tan\frac{\pi}{6} = \frac{PF^2BW^2}{8\sqrt{3}} \qquad (C-7)$$

so that

$$\overline{BSL_2(dB)} = \frac{10}{3} PF^2 \qquad (C-8)$$

The formula just derived should not be used when simulating the search function in a BMD system. The assumption of an average 4 dB beam-shape loss for computing search detection probability, which has been used at times in the past, is also substantially in error. The lattice arrangement of search beams in some cases will result in an overall detection probability higher than the single-hit probability if the target is on the nose of a beam.

As a part of this study, a more refined method of simulating the search function was derived. Detection and measurement correction factors were obtained which should be applied to the beam-nose SJR when computing detection probability and data SJR, respectively. These correction factors have been plotted in Figures C-3 and C-4 for some typical values of PF, false alarm rate, and beam-nose SJR.

The correction factors were computed by a Monte Carlo method, assuming that only the three closest beams from the search lattice are capable of detecting a target. With any RV in the search subvolume being scanned, we can associate three detection probabilities, $P_d(i)$, $i = 1, 2, 3$. The probability of obtaining at least one detection is then

$$P_r \left\{ \text{at least one hit} \right\} = 1 - \prod_{i=1}^{3} (1 - P_d(i)) \qquad \text{(C-9)}$$

This probability will vary, due to beam-shape loss variations, with the target's position in the triangle defined by the centers of the three closest beams (Figure C-1). To remove this variation, an average probability was computed by uniformly distributing 250 points in the triangle and using a Swerling III model to compute the probability of at least one hit. After averaging, a Newton-Raphson method was used in conjunction with the Swerling III curve to determine the SJR which would give the same single-hit probability of detection. This was compared with the true beam-nose SJR to obtain the detection correction factor.

If a detection occurs, we are faced with the problem of determining the most likely SJR of the return, for use in generating measurements. The Monte Carlo technique was also used here, along with the following formula concerning conditional expectations and probabilities:

$$E\left\{ SJR/H \right\} = \sum_{i=1}^{3} E\left\{ SJR/j \text{ hits occur} \right\} \cdot P_r\left\{ j \text{ hits occur}/H \right\} \qquad \text{(C-10)}$$

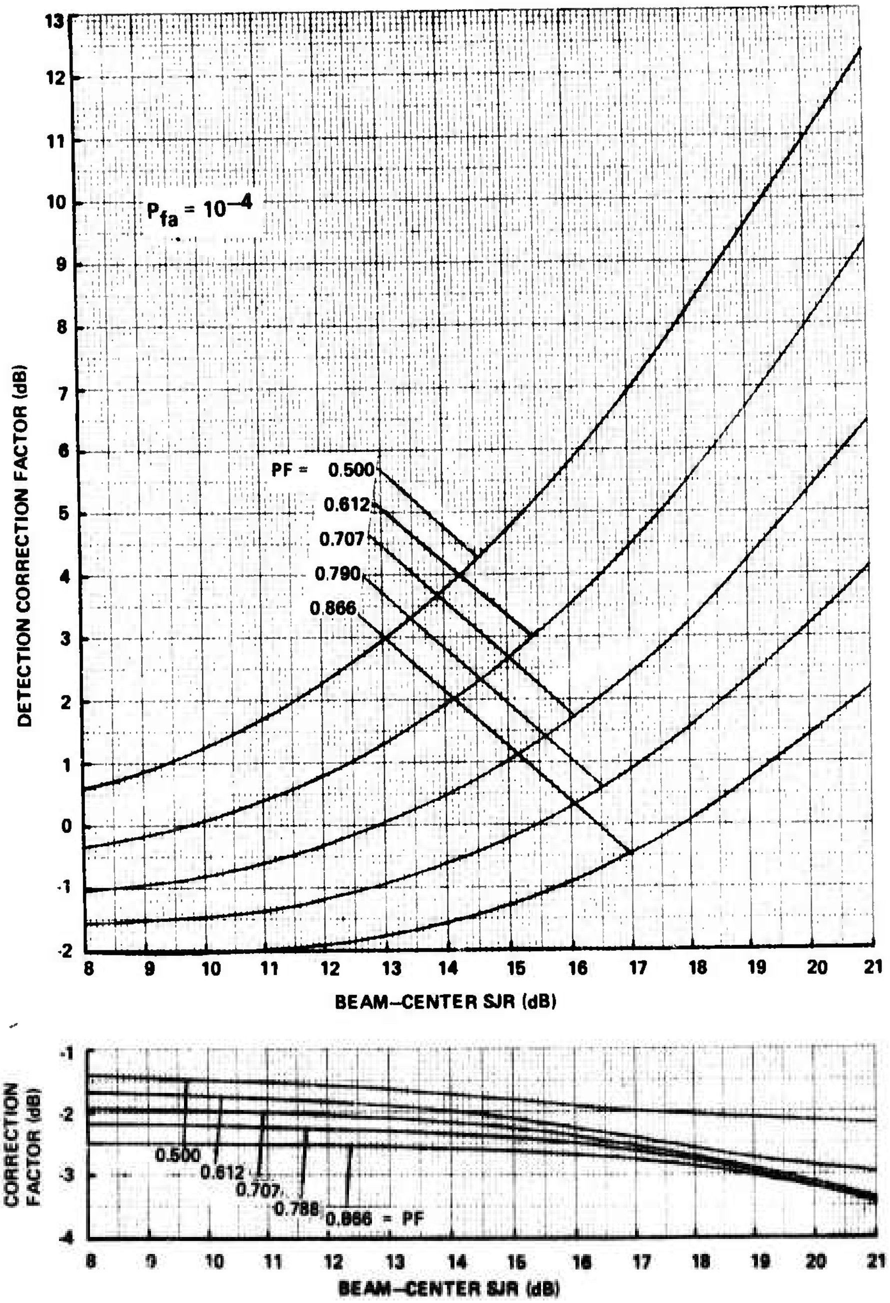


Figure C-3. Correction Factor Sensitivity to Packing Factor and SJR

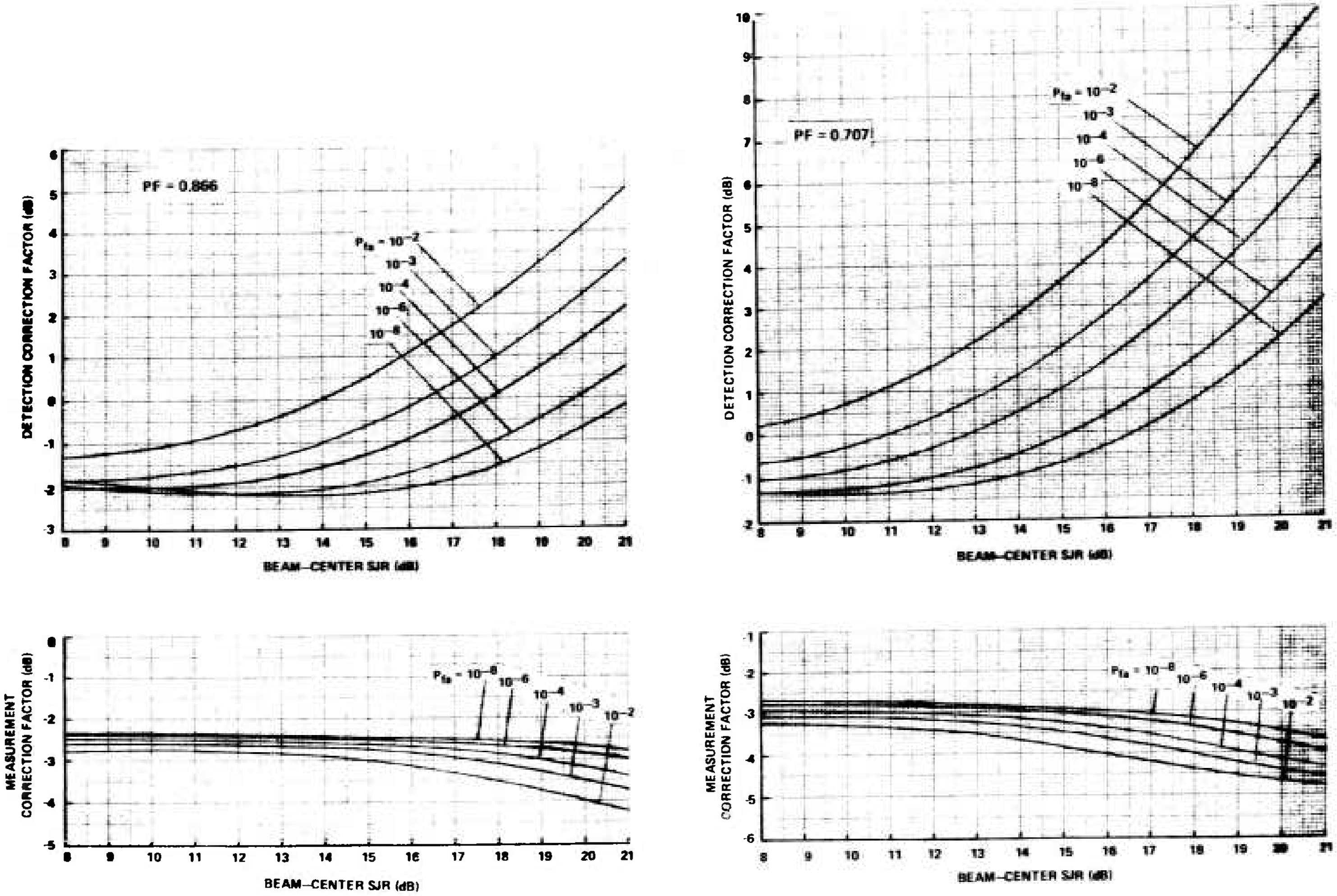


Figure C-4. Correction Factor Sensitivity to False Alarm Rate and SJR (Sheet 1 of 2)

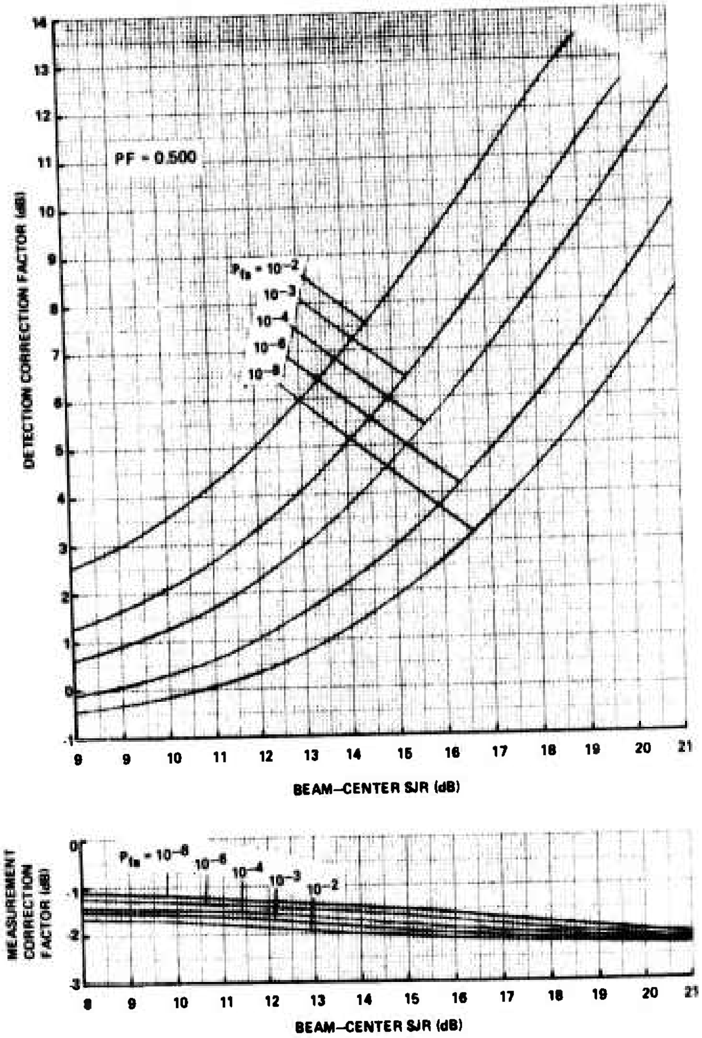


Figure C-4. Correction Factor Sensitivity to False Alarm Rate and SJR (Sheet 2 of 2) where H is the event "at least one hit occurs." For each point in the Monte Carlo simulation, the SJRs of the possible returns were computed and weighted by the detection probabilities as follows:

$$
\begin{aligned}
E\left\{SJR/H\right\} = {} & \sum_{i=1}^{3} SJR(i)P_d(i) \prod_{k\neq i} (1 - P_d(k)) \Big/ 1 - \prod_{j=1}^{3} (1 - P_d(j)) \\
& + \sum_{i=1}^{3} \frac{\sum\limits_{k\neq i} P_d(k)\cdot SJR(k)}{\sum\limits_{k\neq i} P_d(k)} (1 - P_d(i)) \prod_{k\neq i} P_d(k) \Big/ 1 - \prod_{j=1}^{3} (1 - P_d(j)) \\
& + \frac{\sum\limits_{i=1}^{3} P_d(i)\cdot SJR(i)}{\sum\limits_{i=1}^{3} P_d(i)} \prod_{i=1}^{3} P_d(i) \Big/ 1 - \prod_{j=1}^{3} (1 - P_d(j))
\end{aligned}
\tag{C-11}
$$

This expectation, averaged over the random locations, was compared with the true beam-nose SJR to obtain the measurement correction factor.

If these correction factors are used in a simulation, an improvement in search performance will be realized over that obtained by assuming an average 4 dB beam-shape loss in search mode.

APPENDIX D

THE USE OF LINEAR CONSTRAINTS IN THE ADAPTIVE PROCESSOR

D.1 DEVELOPMENT

In its simplest form, the adaptive array processor seeks a solution which optimizes the output SNR with the form of the signal specified but with no other restriction placed on the solution. Frequently, it is desirable to restrict the solution in certain additional ways. An example of this is the placing of a simple constraint on the amplitude of the response of the array in the direction of the desired signal.

The mathematical development which follows is based upon a paper by Frost[9]. We begin with the residue function

$$P_r = W^H M W \qquad \text{(D-1)}$$

We wish to find the value of the vector W which minimizes the residue power, P_r, subject to the constraint

$$C^t W = F \qquad \text{(D-2)}$$

Let the dimension of the vector W be N. Let C be an N x J matrix and F a J-element vector. The columns of C are the constraint vectors. The elements of F represent the inner products of each of the constraint vectors with the weight vector and are assumed to be given.

Using the method of Lagrange multipliers, we first form the cost function including the statement of the constraints.

$$H(W) = \frac{1}{2} W M^H W + \lambda (C^t W - F) \qquad \text{(D-3)}$$

Taking the gradient of H(W) with respect to W and setting it to zero yields

$$W_{opt} = -M^{-1} C^* \lambda \qquad \text{(D-4)}$$

Substituting Equation D-4 into Equation D-2 yields

$$\lambda = -(C^t M^{-1} C^*)^{-1} F \tag{D-5}$$

or finally,

$$W_{opt} = M^{-1} C^* \; (C^t M^{-1} C^*)^{-1} F \tag{D-6}$$

As a simple example, let the gain be held constant in the direction of the desired signal. That is,

$$S^t W = 1 \tag{D-7}$$

which is a specific example of the constraint Equation D-2. For this example, the dimension J = 1 (a single constraint), C = S, and F = 1. Equation D-6 then becomes the familiar

$$W_{opt} = M^{-1} S^* \; (S^t M^{-1} S^*)^{-1} \tag{D-8}$$

It can easily be seen that Equation D-8 satisfies the condition specified in Equation D-7.

A more general example of the application of linear constraints concerns the requirement for quiescent weighting which is other than uniform illumination. Perhaps Dolph-Tchebychev weighting is desired in the absence of jamming, or perhaps a monopulse difference pattern is being employed. In any case, let the desired weighting vector be given by

$$T = aS^* + bR^* \tag{D-9}$$

where a and b are scalar constants and R^* is a unit length vector orthogonal to S^*. That is, the vector T is expressed in terms of mutually orthogonal components S^* and R^*. R^* is given by

$$R^* = \frac{\hat{P}T}{|\hat{P}T|} \tag{D-10}$$

where $\hat{P}$ is the complimentary projection operator

$$\hat{P} = (I - S^* S^t) \tag{D-11}$$

Referring to Equation D-6, the dimension J is two and the columns of C are S and R, respectively. The vector F is seen to be

$$F = \begin{bmatrix} a \\ b \end{bmatrix} \tag{D-12}$$

The weight vector W can, of course, be expanded in any one of an infinite set of orthogonal basis vector sets. If, however, that set is chosen which includes S^* and R^*, then one can see that the solution is constrained such that the coefficients of these components are always a and b. All other changes in W are constrained to directions perpendicular to the plane containing S^* and R^*.

Frost(9) goes on to develop a steepest descent algorithm based upon the same concepts. He shows that the expected value of the correction applied to the weight vector at any step is given by

$$E(W_{k+1} - W_k) = -k \hat{P}M\hat{P} W_k \tag{D-13}$$

where now $\hat{P}$ is the more general form of the complimentary projection operator[†]

$$\hat{P} = I - C^* (C^tC^*)^{-1}C^t \tag{D-14}$$

The modified covariance matrix $\hat{P}M\hat{P}$ can be shown to have exactly J zero eigenvalues corresponding to the J columns of the matrix C^*, assuming that the latter were chosen to be mutually orthogonal in the first place. This means that the increments applied to the weight vector (as in Equation D-13) are constrained to the N-J hyperplane described by the set of vectors which contains no components in the directions described by the columns of C^*.

D.2 IMPLEMENTATION

With reference to Equation D-6, implementation of the direct matrix inversion algorithm using constraints would involve the inversion of the J x J

[†] If the columns of C are unit length mutually orthogonal vectors, $C^TC^* = I$.

matrix $C^t M^{-1} C^*$ for each beam formed (since C is different for each beam). For one, two, or even three constraints, it is conceivable to write out the general solution for the inverse, thus avoiding the requirement of computing a matrix inverse for each case.

The Frost form of the steepest descent algorithm is designed to control the accumulation of errors (due to roundoff, etc.) in the determination of the constrained solution. The iteration algorithm is given by

$$W_{k+1} = \hat{P} \, (I - \mu \, V^* V^t) \, W_k + \psi \qquad \text{(D-15)}$$

where $\hat{P}$ is given by Equation D-14 and ψ is given by

$$\psi = C^* \, (C^T C^*)^{-1} F \qquad \text{(D-16)}$$

The process is started with the initial condition

$$W_0 = \psi \qquad \text{(D-17)}$$

Noting that the factor $V^t W_k$ is the residue v_{rk} for the k'th step, the computation for each step involves the multiplication by the matrix $\hat{P}$ of the vector $W_k - v_{rk} V^*$ and a vector addition of ψ. Unless P is of a simple form, this involves N^2 multiplications per iteration (or N vector multiplications) which is somewhat more complex than was the unconstrained algorithm.

APPENDIX E

DERIVATION OF MATRIX INVERSE RECURSIVE UPDATE RELATIONSHIP

In the real-time processing of data for adaptive arrays, one wishes to perform the following operation repeatedly for each new sample for data:

$$M_{i+1}^{-1} = (\alpha M_i + \beta V_i^* V_i^t)^{-1} \tag{E-1}$$

The matrix M_i can always be represented in terms of the sum of diadics.

Consider the following development which leads to a result of the form

$$M_{i+1}^{-1} = M_i^{-1} - A \tag{E-2}$$

Start with the expression

$$(\alpha M_i + \beta V_i^* V_i^t)(\frac{1}{\alpha} M_i^{-1} - A) = I \tag{E-3}$$

Expanding this equation and dividing through by α yields

$$\frac{\beta}{\alpha^2} V_i^* V_i^t M_i^{-1} - \frac{\beta}{\alpha} V_i^* V_i^t A - M_i A = 0 \tag{E-4}$$

With some insight into the form of the desired result, we make the following hypothesis of the form of the matrix A:

$$A = \mu M_i^{-1} V_i^* V_i^t M_i^{-1} \tag{E-5}$$

This expression is substituted into Equation E-4 and the terms involving μ are grouped, noting that the quadratic form $V^t M^{-1} V^*$ is a scalar.

$$\mu \left[\frac{\beta}{\alpha} (V_i^t M_i^{-1} V_i^*) + 1 \right] V_i^* V_i^t M_i^{-1} = \frac{\beta}{\alpha^2} V_i^* V_i^t M_i^{-1} \tag{E-6}$$

from which

$$\mu = \frac{\beta}{\alpha^2} (1 + \frac{\beta}{\alpha} V_i^t M_i^{-1} V_i^*)^{-1} \tag{E-7}$$

or finally,

$$M_{i+1}^{-1} = \frac{1}{\alpha} M_i^{-1} - \mu M_i^{-1} V_i^{*} V_i^{t} M_i^{-1} \qquad \text{(E-8)}$$

One must now intelligently select the values of the constants α and β. To obtain a form of M_K which is a simple average over K samples, one chooses

$$\alpha = \frac{K}{1 + K} \quad , \quad \beta = \frac{1}{1 + K} \qquad \text{(E-9)}$$

This form is suitable for processing finite blocks of data since, for large K, the most recent samples have smaller and smaller effect on the solution.

If one wishes to process continuously and under quasistationary (slowly changing) conditions, a better form consists of

$$\alpha = \rho \ , \quad \beta = 1 - \rho \ , \quad \rho < 1 \qquad \text{(E-10)}$$

This has the effect of multiplying the K-J'th sample by ρ^J so that old data is gradually "forgotten" in an exponentially decaying fashion.

APPENDIX F

CONVERGENCE FOR THE DIRECT MATRIX INVERSION TECHNIQUE

In a recent paper by Reed,$^{(6)}$ there is developed a relationship which bounds the number of samples required to form an acceptable estimate of the sample covariance matrix. A brief outline of that development will be given here.

For arbitrary weight W, desired signal S, and true covariance matrix M, the SNR at the output is given by

$$SNR = S^t W\ W^H S^* \ (W^H M\ W)^{-1} \qquad \text{(F-1)}$$

The theoretical optimum weighting is

$$W_{opt} = a\ M^{-1} S^* \qquad \text{(F-2)}$$

Substituting Equation F-2 into Equation F-1 yields

$$SNR_{opt} = S^t M^{-1} S^* \qquad \text{(F-3)}$$

Let ρ be the ratio of Equation F-1 to Equation F-3, that is, a measure of the optimality of an arbitrary weight vector, W.

$$\rho = SNR_W / SNR_{opt} \qquad \text{(F-4)}$$

For the direct matrix solution, the estimated optimum weight vector, $\hat{W}$, is given by

$$\hat{W} = a\ \hat{M}^{-1} S^* \qquad \text{(F-5)}$$

where $\hat{M}$ is the sample covariance matrix.

Combining Equations F-1 and F-3 through F-5, the expression for ρ becomes

$$\rho\ \hat{W} = (S^t \hat{M}^{-1} S^*)^2 (S^t \hat{M}^{-1} M\ \hat{M}^{-1} S^*)^{-1} (S^t M^{-1} S^*)^{-1} \qquad \text{(F-6)}$$

The sample covariance matrix is given by

$$\hat{M}_K = \frac{1}{K} \sum_{i=1}^{K} V_i^* V_i^t \qquad \text{(F-7)}$$

The statistical behavior of $\hat{M}_K$ can be described by a complex Wishart distribution, CW(K, N, M), with parameters: K, the number of samples; N, the order of the matrix; and M, the true covariance matrix. Using this and the fact that $C^H \hat{M} C$ (where C is any non-singular transformation) is also CW(K, N, $C^H M$ C), Reed uses the previous work of Goodman and Capon to derive the first and second moments of ρ.

$$E(\rho) = (K + 2 - N)/K + 1) \qquad \text{(F-8)}$$

and

$$E(\rho^2) = \frac{(K + 2 - N)(K + 3 - N)}{(K + 2)(K + 1)} \qquad \text{(F-9)}$$

APPENDIX G

BOUNDS ON THE CONTROL LOOP TIME CONSTANT

The smoothed control equation for the weights in an analog control loop implementation of the adaptive processor can be shown to be

$$W(t) = S^{*} - h(t)\ (*)\ G\ M\ W\ (t) \tag{G-1}$$

where S is the desired signal response, h(t) is the filter impulse response, G is a diagonal matrix of electronic gains, and (*) denotes convolution. The filter generally used is the simple low-pass filter given by

$$H(s) = 1/(1 + T_I s) \tag{G-2}$$

where H(s) is the Laplace transform of the impulse response h(t). Making the standard transformation to "normal" or decoupled coordinates using the modal matrix transformation A (see Section 5), one obtains

$$Z = Y - h(t)\ (*)\ K\ Z \tag{G-3}$$

where

$$W = A\ Z \tag{G-4}$$

$$Y = A^{H} S^{*} \tag{G-5}$$

and the diagonal matrix of eigenvalues, k_i, is given by

$$K = A^{H} G^{1/2} M\ G^{1/2} A \tag{G-6}$$

The expression for the i'th normal mode weight, z_i, is given by

$$z_i(t) = \frac{y_i}{1 + k_i} \left[1 + k_k \exp(- \frac{t(1 + k_i)}{T_I}) \right] \tag{G-7}$$

or the enhanced time constant for the i'th mode is

$$T_{ei} = \frac{T_I}{1 + k_i} \tag{G-8}$$

In order to bound the time constant T_{ei}, it is necessary to examine Equation G-6 in more detail. Assume that the processor consists of N identical channels, each containing equal amounts of uncorrelated thermal noise with power P_n and correlated jamming with power P_j. The gain matrix, G, simplifies to the scalar g given by

$$g = Q\ /(P_n + P_j)^{1/2}\ P_n^{1/2} \tag{G-9}$$

assuming that a hard limiter is employed in the loop (without the limiter, g = Q). Q is a circuit constant. The matrix M is given by

$$M = P_n I + P_j\ R \tag{G-10}$$

where I is the identity matrix and R contains the normalized correlation coefficients ρ_{ij}. Substituting Equations G-9 and G-10 into Equation G-6 yields

$$K = \frac{Q}{(1 + P_j/P_n)^{1/2}} \left[I + \frac{P_j}{P_n} A^H R\ A \right] \tag{G-11}$$

The trace of R is N. Similarly, the trace of $A^H R\ A$ is also N (the latter matrix being diagonalized). For the simple case of one narrowband jammer, there is only one non-zero element in $A^H R\ A$ and it's value is therefore N. For more complex situations, there are more non-zero elements but the sum (or trace) is N. Thus, the bounds on k_i are

$$k_{min} = Q\ /(1 + P_j/P_n)^{1/2} \tag{G-12}$$

and

$$k_{max} = k_{min}\ (1 + N\ P_j/P_n) \tag{G-13}$$

Substituting these relationships into Equation G-8 yields

$$Te_{max} = T_I \tag{G-14}$$

and

$$Te_{min} = \frac{T_I}{1 + Q\ N\ (P_j/P_n)^{1/2}} \qquad \text{(G-15)}$$

There is a practical limit on the value of the time constant, T_I, which places a lower bound on it. If the enhanced bandwidth (inversely proportional to the enhanced time constant, T_e) of a loop exceeds more than about 10% of the signal processing bandwidth B_s, the weights become noisy and contribute to increased noise residue. Thus, the limit on T_I, the filter time constant, is

$$T_{I_{min}} = 10\ Q\ N\ (P_j/P_n)^{1/2}/(\pi B_s) \qquad \text{(G-16)}$$

In summary, then, the time constants of a system (which has been adjusted according to Equation G-16) can be expected to fall in the range of

$$10/(\pi B_s) \leq T_e \leq 10\ Q\ N\ (P_j/P_n)^{1/2}/(\pi B_s), \qquad \text{(G-17)}$$

Typically, Q is 3.16. For N = 100 and a 40 dB JNR, the range of time constants is 3×10^4:1.

APPENDIX H

AN ATTEMPT AT A STATISTICAL ESTIMATE OF CONVERGENCE TIME FOR THE STEEPEST DESCENT ALGORITHM

In Appendix F, it is shown that the convergence rate of the direct matrix algorithm can be estimated using the statistical properties of the sample covariance matrix. It is desired to make the same estimate for the steepest descent algorithm so that the two algorithms can be compared on the same basis. Because the analysis in this appendix is incomplete, only the initial steps of the development are indicated.

Consider the simplest form of the steepest descent algorithm:

$$W_{i+1} = (I - kV_i^* V_i^t) W_i \qquad \text{(H-1)}$$

If the initial value of W is

$$W_1 = S^* \qquad \text{(H-2)}$$

then, after K iterations the result is

$$W_{K+1} = \prod_{i=1}^{K} (I - kV_i^* V_i^t) S^* \qquad \text{(H-3)}$$

Expanding, one obtains

$$W_{K+1} = (I - kV_1^*V_1^t)(I - kV_2^*V_2^t)(....)(I - kV_K^*V_K^t)S^* \qquad \text{(H-4)}$$

or

$$W_{K+1} = I - kK \hat{M}_K + \sum_{i=1}^{K} kV_i^*V_i^t (kK \hat{M}_K - kV_i^*V_i^t)$$

$$+ .... +(-1)^K kK \prod_{i=1}^{K} V_i^*V_i^t S^* \qquad \text{(H-5)}$$

In other words, the weight can be expressed in terms of a rather complex function of the sample covariance matrix, $\hat{M}_K$, given by

$$\hat{M}_K = \frac{1}{K} \sum_{i=1}^{K} V_i^* V_i^t \qquad \text{(H-6)}$$

Using the method o. Appendix F, the ratio of the signal-to-noise ratio due to arbitrary weight, W, to optimum signal-to-noise ratio is

$$\rho_W = S^t W \, W^H S^* \, (W^H M \, W)^{-1} (S^t M^{-1} S^*)^{-1} \qquad \text{(H-7)}$$

where S is the desired signal vector and M is the true covariance matrix. The straightforward approach is now to substitute Equation H-5 into Equation H-7 and find the expected value of ρ_W as a function of K and the order, N.

This analysis is incomplete as presented because the derivation of $E(\rho)$ is apparently a very complex process. It is hoped that the analysis can be completed in the next phase of the contract.

DISTRIBUTION LIST

Addressee	Copies
Director U.S. Army Ballistic Missile Defense Program Office 12th Floor, Commonwealth Building 1300 Wilson Boulevard Arlington, Virginia 22209 ATTN: DACS-BMT	1
Director U.S. Army Ballistic Missile Defense Advanced Technology Center P.O. Box 1500 Huntsville, Alabama 35807 ATTN: Mr. W. Dickinson (ATC-R)	1
Naval Research Laboratory Washington, D.C. 20375 ATTN: Code 526 (Dr. W. Gabriel)	1
General Research Corporation Washington Operations 1501 Wilson Boulevard, Suite 700 Arlington, Virginia 22209 ATTN: Mr. R. Yost	1
Calspan Corporation P.O. Box 235 Buffalo, New York 14221 ATTN: Mr. I. Sacks	1
Defense Documentation Center Cameron Station Alexandria, Virginia 22314	2
Massachusetts Institute of Technology Lincoln Laboratory P.O. Box 73 Lexington, Massachusetts 02173 ATTN: Mr. V. Guethlen	1
Massachusetts Institute of Technology Lincoln Laboratory P.O. Box 73 Lexington, Massachusetts 02173 ATTN: Technical Library	1
Syracuse University Research Corporation Merrill Lane, University Heights Syracuse, New York 13210 ATTN: Mr. D. Morezak	25

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER
SURC TR 75-249 (Volume 6) ✓

2. GOVT ACCESSION NO.

3. RECIPIENT'S CATALOG NUMBER

4. TITLE (and Subtitle)
ECM/ECCM Evaluation Program
Large Aperture Adaptive Array Study

5. TYPE OF REPORT & PERIOD COVERED
Final Report
1/29/74 - 2/28/76

6. PERFORMING ORG. REPORT NUMBER

7. AUTHOR(s)
Dean J. Chapman
Stephen G. Kamak

8. CONTRACT OR GRANT NUMBER(s)
DAHC60-73-C-0044 ✓

9. PERFORMING ORGANIZATION NAME AND ADDRESS
Syracuse University Research Corporation ✓
Merrill Lane, University Heights
Syracuse, New York 13210

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS

11. CONTROLLING OFFICE NAME AND ADDRESS
U.S. Army Ballistic Missile Defense
Systems Command, P.O. Box 1500
Huntsville, Alabama 35807, ATTN: BMDSC-CRS

12. REPORT DATE
March 1976

13. NUMBER OF PAGES
100

14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)
U.S. Army Ballistic Missile Defense
Advanced Technology Center,
P.O. Box 1500, Huntsville, Alabama 35807
ATTN: BMDATC-R

15. SECURITY CLASS. (of this report)
UNCLASSIFIED

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)
Distribution limited to U.S. Government Agencies only; Test and Evaluation; 4 October 1974; other requests for this document must be referred to Ballistic Missile Defense Advanced Technology Center, ATTN: ATC-2, P.O. Box 1500, Huntsville, Alabama 35807.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)
Electronic Countermeasures
Phased Array Radar
Adaptive Apertures

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)
This report is concerned with the problem of protecting a large phased array radar from the effects of electronic countermeasures (ECM). The context of the problem is the defense of strategically important targets from ballistic missile attack, the radar being one of the key sensors in the defense system.

A continuation of previously reported work, this study investigated the feasibility of using a partially adaptive array to emulate, as nearly as

possible, the performance of the fully adaptive array. A partially adaptive array would be less expensive to fabricate and less complex and, therefore, would be capable of more rapid response.

This study was divided into three parts. First, since the partially adaptive array can be implemented in many different configurations, several of the more promising of these configurations were investigated. Second, the fundamental question of transient response was addressed. The findings of this portion of the study apply to any adaptive processor. A potential solution to the transient response problem was postulated and evaluated. Since this solution, which primarily addressed problems arising during the radar search function, had a potential effect on the performance of the radar during the track function, a third and much shorter study was launched into the question of tracking performance of the array.

The results of these studies have shown that there are several partially adaptive array configurations which show promise of providing the protection needed against sidelobe jamming. Further, response time of these arrays can be controlled to within acceptable limits. There does not, however, appear to be a simple solution to the requirement for precision in the fabrication of the partially adaptive array. Array errors would seem to pose a fundamental limitation to this approach.